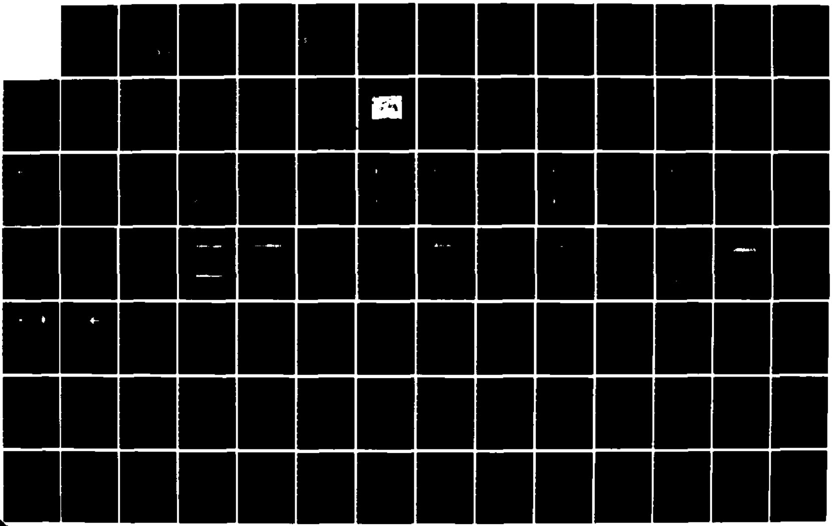
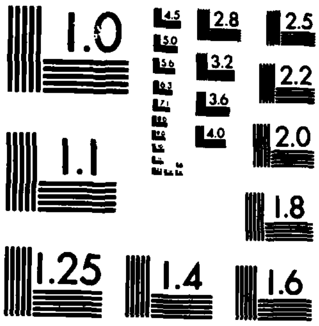
AD-A150 170 EMPLOYMENT OF ADAPTIVE LEARNING TECHNIQUES FOR THE DISCRIMINATION OF ACOU..(U) GENERAL ELECTRIC CORPORATE RESEARCH AND DEVELOPMENT SCHENECTA.. J W ERKES ET AL. 1/2

UNCLASSIFIED DEC 84 84SRD002 N00014-82-C-2031 F/G 20/1 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

(12)

EMPLOYMENT OF ADAPTIVE LEARNING TECHNIQUES FOR THE DISCRIMINATION OF ACOUSTIC EMISSIONS

J.W. Erkes and K.C. Tam
General Electric Corporate Research and Development

S.R. Mannava
General Electric Turbine Technology Laboratory

J.F. MacDonald and H.A. Scarton
Rensselaer Polytechnic Institute

PHASE II FINAL REPORT
Contract N00014-82-C-2031

December 1984

Prepared by

J.W. Erkes, Project Manager
General Electric Company
Corporate Research and Development
Schenectady, New York 12345

Prepared for

H. Chaskelis, Project Manager
Naval Research Laboratory
4555 Overlook Avenue, S.W.
Washington, DC 20375

84SRD002

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

AD-A150170

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION: Unclassified	1b. RESTRICTIVE MARKINGS: None
2a. SECURITY CLASSIFICATION AUTHORITY: N/A	3. DISTRIBUTION/AVAILABILITY: Approved for public release; Approved for Distribution: Distribution Unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): 84SRD002	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION: General Electric Company; 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State and ZIP Code): General Electric Company, Corporate Research and Development, 1 River Road, Schenectady, NY 12345	7b. ADDRESS (City, State and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION: Naval Research Laboratory; 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: N00014-82-C-2031
8c. ADDRESS (City, State and ZIP Code): 4555 Overlook Ave., S.W. Washington, DC 20375	10. SOURCE OF FUNDING NOS.: PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT NO.
11. TITLE (Include Security Classification): Employment of Adaptive Learning Techniques for the Discrimination of Acoustic Emissions (Unclassified)	

12. PERSONAL AUTHOR(S): Erkes, J.W.; Tam, K.C.; Mannava, S.R.; MacDonald, J.F.; Scarton, H.A.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Yr., Mo., Day)	15. PAGE COUNT
Phase II Final Report	FROM Nov 82 TO Nov 83	December 1984	

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB. GR.	Acoustic Emission, Digital Signal Processing, Adaptive Learning, Homomorphic Deconvolution,

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Under Phase I of this contract, several new acoustic emission techniques were developed. These techniques showed promise as a means of adaptively learning and removing multimode and multipath effects from acoustic emission signals in a preprocessing step prior to source characterization. In this Phase II effort, software was developed to implement the techniques, and a series of experiments was carried out to assess the effectiveness and practicality of these techniques on real signals. Analysis of the data revealed that reasonably accurate transfer functions could be determined adaptively, when a flat response wide-band transducer was employed, and when relatively short record lengths were used. Available commercial transducers with the requisite frequency response are quite fragile, however, and are far too insensitive for practical use. Pattern recognition techniques were also explored as a means of characterizing acoustic emission sources; specifically, details of the pulse microstructure were examined for evidence of characterizable features. No such evidence was found in either "raw" (reverberation dominated) or processed data. Originator supplied keywords included -

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT: UNCLASSIFIED/UNLIMITED X SAME AS RPT. ☐ DTIC USERS ☐	21. ABSTRACT SECURITY CLASSIFICATION: Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL: H. Chaskelis	22b. TELEPHONE NUMBER (Include Area Code): (202) 767-3613; 22c. OFFICE SYMBOL

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont'd)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (Cont'd)

LIST OF ILLUSTRATIONS (Cont'd)

Section 1

INTRODUCTION AND SUMMARY

1.1 Introduction

Until recently, workers in the acoustic emission (AE) research community appear to have been unaware of a number of techniques applied successfully in other research areas (including sonar, seismology, and speech processing) where certain advanced signal processing techniques are frequently used to deal with distortion effects imposed on the signals of interest. For example, volume reverberation and multimode propagation effects are common problems in these applications, and much effort has been expended in the development of advanced techniques to compensate for them. Since these effects also severely distort real AE signals, those advanced techniques are expected to be useful in improving the performance and reliability of AE-based nondestructive evaluation (NDE) systems. This report describes the results of a research program that seeks to capitalize on those successful research efforts, extending them and applying them to the problems of the detection, location, and characterization of AE events.

1.2 Signal Processing

Current AE analysis techniques make limited use of the phase information in the received signal and rely on a variety of incoherent techniques for source identification and location; pulse energies, rise times, ringdown times, and amplitude distributions are commonly used signal parameters. Although these parameters often yield useful information about the nature and location of the AE source, they do have serious limitations. In particular, they tend to be less useful when the signal-to-noise (S/N) ratio is poor, when complex structural features produce confusing reflections, or when multiple sources are present. Unfortunately, the most important application areas for acoustic emissions often suffer badly from just these problems. Specifically, on-line AE-based NDE systems often must deal simultaneously with adverse S/N ratios, complex structures producing serious multipath/multimode interference, as well as unknown (often multiple) source locations.

Coherent methods (which make use of the phase information) provide some basis for hope in dealing with these complex problems. Unfortunately, linear methods to compensate for these effects have not proven to be particularly useful. In light of the role of mode conversion and multimode propagation in signal distortion, this is perhaps not surprising. Nonlinear methods offer considerably more promise and, in fact, have been used with much success in dealing with similar problems in other fields. Homomorphic deconvolution especially has been used with considerable success to eliminate multipath and multimode distortion in seismic, speech, and audio processing applications where S/N ratios are relatively good. A series of potentially useful homomorphic signal processing approaches were developed under Phase I of this contract and are described in detail in the Phase I final report. This report will focus on the experimental evaluation of the potential use these techniques may ultimately find in practical acoustic emission problems and, in particular, evaluate their

potential for compensating for transducer and structural resonances. Reliable multimode and multipath compensation, if achievable, can provide the technical basis for automated on-line AE monitoring of complex structures for cracks; if the structural complexities can be reduced or eliminated, it may be practical to monitor crack growth rates, to carry out crack severity assessments, and to locate cracks.

In other situations, particularly when the S/N ratio is poor, homomorphic methods, especially those based on cepstral techniques, are less useful and suffer from problems with accurate phase unwrapping. Under these conditions, adaptive nonlinear methods may be very applicable. The experimental work reported here focused on the deconvolution problem for situations where a relatively good S/N ratio was available.

1.3 Sensor Evaluation

The most commonly used AE transducers—resonant piezoelectric transducers—seriously distort the microstructure of AE signals through transducer ringdown. Although the nonlinear signal processing approaches outlined above will to a large extent compensate for those problems, a better solution would be to use a transducer without massive intrinsic phase distortion. In this project we studied the performances of both the resonant transducers and the conical transducers. The conical transducers used were commercially available piezoelectric sensors based on ideas developed and demonstrated by Eitzen et al.[1] at the National Bureau of Standards. They have relatively flat frequency response up to 1 MHz. Such improved frequency response is achieved by reducing the contact area of the active element, by creating a smooth acoustical impedance transition to the backing material, and by increasing the backing material volume. Because of the reduced contact area, however, the detected signals are much reduced in amplitude compared to the resonant transducers, and very poor S/N ratios can be a serious problem in practical experimental situations.

1.4 Experimental Confirmation

Finally, as the major element of the Phase II contract effort, a series of experiments of gradually increasing complexity were planned and performed to confirm and evaluate the effectiveness of the signal processing algorithms and techniques developed. The experiments were carried out by General Electric at the Research and Development Center and at the Materials and Processes Laboratory, in Schenectady, N.Y., under the joint direction of the co-investigators.

1.5 Summary

This project represents a radically new approach to the acquisition and analysis of AE signals. The approach is based on the use of new signal processing methods and sensors and focuses on the removal of multipath and multimode distortion from the signals.

Section 2

THE THEORY REVISITED

The analytical basis for this work is described in detail in the Phase I final report, and will not be repeated here. It should be noted, however, that in the course of carrying out the experimental portion of Phase II, it was discovered that some of the theoretical results developed in Phase I of this project needed to be extended, and, in some instances, modified. Some of these changes came about in the course of analyzing the experimental data and in learning more about actual experimental uncertainties. Other changes arose out of a more careful examination of the physical assumptions behind the application of these techniques to the AE problem. The topics modified or added include exponential weighting, bandpass mapping, and the proper choice of the time distribution function in analyzing the multiple-event waveforms. Detailed descriptions of these changes, and the reasons behind them, are outlined below.

2.1 Generalized Time Distribution Function for Multiple-Event Waveforms

As was shown in the Phase I report, the measured signals $y(t)$ in acoustic emission experiments can be written in the form

$$y(t) = \left\{ \left[\sum_i a_i h_{pi}(t-\tau_i) \right] * h_r(t) \right\} \cdot g(t) + n(t) \tag{1}$$

where

$*$ = the operation of convolution

$h_r(t)$ = the ringdown impulse response due to the boundary reverberation and the transducer response

$h_{pi}(t-\tau_i)$ = the AE impulse pulse emitted at the source at time τ_i

$g(t)$ = the measurement time gate

$n(t)$ = the background

The duration of the measurement time gate $g(t)$ is assumed to be much longer than those of h_{pi} and h_r and therefore will be omitted from the rest of this report. This point will be treated in more detail is Section 4.2.

2.2 Single-Event Approximation

Single events represent a much simpler case, and in the case of a solitary event, Equation 1 reduces to

$$y(t) = h_{pi}(t) * h_r(t) + n(t) \tag{2}$$

If the noise component $n(t)$ is negligible, Equation 2 can be treated by the conventional homomorphic deconvolution technique to recover the signal $h_{pi}(t)$. Under this approach, the measurement $y(t)$ is transformed to cepstral domain by the homomorphic operation IFT(LOG(FT($y(t)$))). This composite operation converts the convolution operation into the addition operation. The cepstrum of $y(t)$ is the sum of

those of $h_{pi}(t)$ and $h_r(t)$, and hence the two cepstra can be separated by linear filtering.

This technique will work especially well when the cepstra of $h_{pi}(t)$ and $h_r(t)$ are well separated in cepstral period. In AE measurements, these conditions typically are met; typical AE h_{pi}s are narrow impulses with durations on the order of a few microseconds, while the h_rs have substantial low-frequency energy components and are much longer in duration. Since the cepstrum of a delta-like function is also a delta-like function, the cepstra of the h_{pi}s are concentrated near the origin in the cepstral domain.[2,3] In general the cepstrum of h_r is more spread out; so the criterion of separability of cepstra is approximately satisfied. One way to further reduce the contribution of the cepstrum of h_r near the origin is to convert h_r into a minimum-phase sequence by using the technique of exponential weighting, which will be described in detail in Section 2.4.

Once the cepstra of $h_{pi}(t)$ and $h_r(t)$ are separated, they can be operated on by the inverse homomorphic transform to yield either $h_{pi}(t)$ and $h_r(t)$, respectively. The recovered pulse shape $h_{pi}(t)$ can be analyzed for features associated with the particular class of events characterizing the source.

2.3 Multiple Events

In general, AE events do not occur singly, but rather in associated bursts; in some structures, the events may be easily visible as single pulses, but in large reverberant structures, the individual pulses, extended by reverberation, will overlap and be difficult to distinguish from one another.

2.3.1 Multiple Events with Occasional Single Events

On the other hand, provided that the AE events are associated with a localized crack formation and are statistically stationary, and further provided that a single reverberated AE waveform is available for analysis, the derived ringdown function $h_r(t)$ can be used to dereverberate the multiple-event waveforms (if they occur) from the same source, through simple Fourier inversion. These dereverberated waveforms would reveal the original impulse time sequence of the overlapping individual events, and hence similar data from another transducer located at a different spatial location could be analyzed jointly via cross-correlation techniques to yield information on the time of arrival and hence location of the events. Similarly, the rate of occurrence of these waveforms could be easily determined and could be used in estimating the activity strength and otherwise characterizing the source.

2.3.2 Overlapping Multiple Events

More generally, in dealing with multiple-event waveforms, one has to resort to Equation 1. Compared to Equation 2, Equation 1 has two complications: the constancy of the impulse shape h_{pi} and the time distribution of the events. Assuming the different events all have the common pulse shape $h_p(t)$ and only differ in their amplitudes a_i, then we have

$$y(t) = h_p(t) * p(t) * h_r(t) + n(t) \tag{3}$$

where $h_p(t)$ is the AE pulse emitted at time t, and $p(t) = \sum_i a(i)\delta(t-\tau_i)$ is the time

distribution of the events weighted with amplitudes corresponding to those of the different events.

The cepstrum of the measured signal $y(t)$ is then the sum of the cepstra of $h_p(t)$, $p(t)$, and $h_r(t)$. If the pulse train $p(t)$ has been converted to minimum phase, it can been shown that its cepstrum is nonzero only for positive times greater than or equal to the interval between the first two arrivals,[2] and thus is well separated from that of $h_p(t)$ if the intervals between the impulses are not too close to each other. In this case, $h_p(t)$ can again be recovered by homomorphic deconvolution.

(If the different events do not have the same pulse shape, the signals $\sum_i a_i h_{pi}(t-\tau_i)$ contained in the measurement $y(t)$ cannot be simplified to a convolution of the pulse train $p(t)$ with a common pulse shape $h_p(t)$, as in Equation 3. Since it is difficult to estimate the cepstra of such complex signals, cepstral analysis is not practical.)

On the other hand, since the cepstra of $h_r(t)$ and $p(t)$ usually overlap, $h_r(t)$ cannot be separated out. (Note that this result is distinctly different from the result reported in the Phase I study, where the assumption that an exponential distribution function holds for an ensemble of overlapping AE pulses lead to a contrary conclusion. A more careful analysis of the physical situation suggests that an exponential distribution is not appropriate for an overlapping ensemble of AE pulses.) Fortunately there is another way out of this dilemma. The method works on the fact that if $p(t)$ is in fact a delta-like function, then the cepstrum of $p(t)$ is also a delta-like function and thus is well separated from that of $h_r(t)$. One way to simulate this situation is by averaging a large number of waveforms with one of the events in each waveform aligned at about the same location. A simple and practical way of accomplishing this, for example, is to use a transient waveform capturing device set to trigger on a strong impulse. This will result in a series of captured waveforms where the first strong event in each waveform which exceeds the triggering level always occurs at about the same location t_0 in the waveform records. Under these circumstances, the time distribution function can be approximated by

$$p(t) = \delta(t - t_0) + b \tag{4}$$

where b represents the averaged time distribution of the other events in the waveforms.

The procedure is illustrated in Figure 1. If the number of waveforms N used in the averaging is large enough, b will approach a uniformly flat distribution. The magnitude of b decreases with the number N, approaching zero as N tends to infinity. In this case $p(t)$ is basically a delta function with a cepstrum concentrated at the origin, and thus will not interfere with the cepstrum of $h_r(t)$.

Note that if there were no aligning in the averaging, all the impulses would be distributed randomly in location, and $p(t)$ would approach a uniformly flat distribution. Since the convolution of a flat distribution with any function is also a flat distribution, the averaged waveform $y(t)$ is also a flat distribution and therefore contains no information at all.

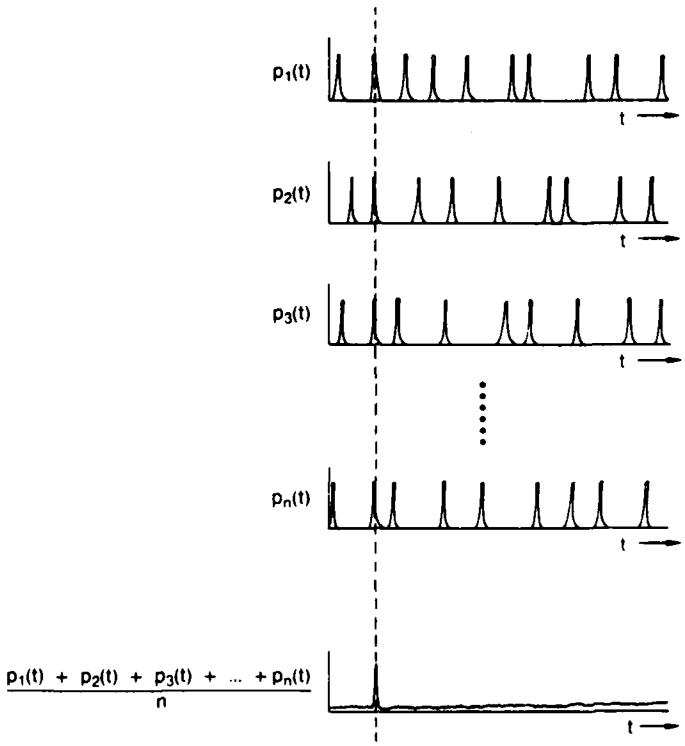


Figure 1. The delta-shaped time distribution function obtained by averaging the time distribution function of a large number of multiple-event waveforms with one of the events in each waveform aligned

2.4 Exponential Weighting

One of the steps in cepstral analysis is phase unwrapping, which is the process of making the phase θ of the Fourier transform, $r\exp(i\theta)$, of the time series waveform $\{a(n)\}$ into a continuous function.[2] However, if some Fourier components are zero, their phase θ would be undefined, and phase unwrapping would fail. This problem is especially serious with waveforms of long record length. The reason for this is that long record length corresponds to fine sampling in the Fourier domain,[4] and hence waveforms with long record length are more likely to pick up zero Fourier components than those with short record length. This phenomenon is illustrated in Figure 2.

Now the Fourier transform of a waveform corresponds to the z-transform of the function at the unit circle $|z| = 1$, where the z-transform $A(z)$ of a sequence $a(n)$ is defined as:

$$A(z) = \sum_{n=\infty}^{\infty} a(n) z^{-n},$$

where z is a complex variable.

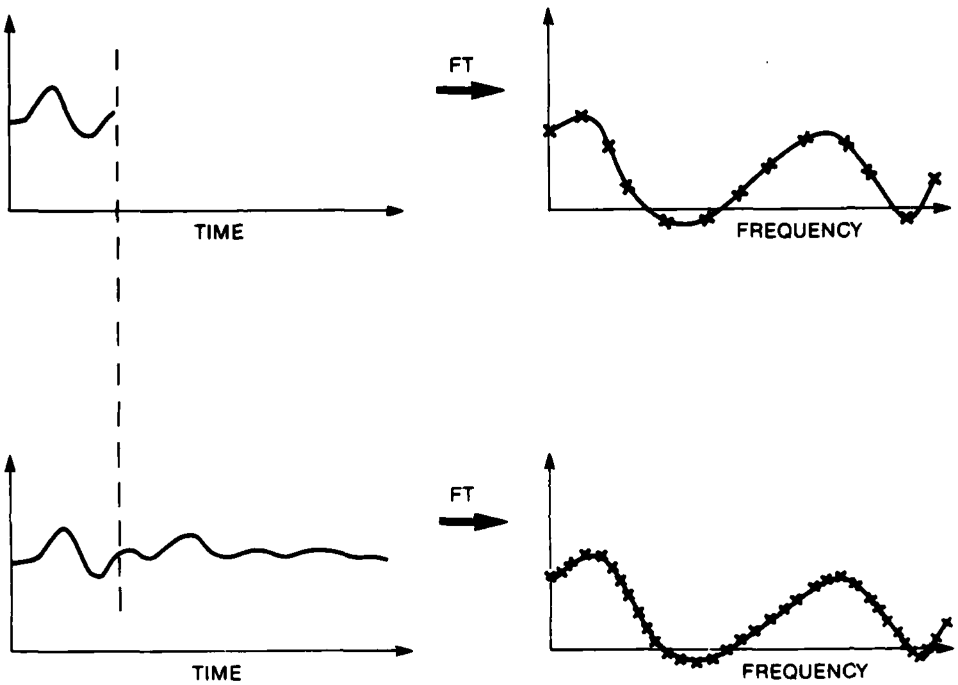


Figure 2. The relation between record length and frequency sampling interval

One way to get around the phase wrapping problem when some Fourier components are zero is by using exponential weighting.[2] The original waveform $\{a(n)\}$ is weighted with the series $\{\alpha^n\}$ before z-transformation:

$$\{a(n)\} \rightarrow \{a(n)\alpha^n z^{-n}\}$$

where α is a real number less than 1. Exponentially weighting a function by $\{\alpha^n\}$ has the effect of scaling the magnitude of the zeroes of the z-transform of the function by α, as illustrated in Figure 3. Thus the zeroes in the Fourier transform of a function can be removed by exponentially weighting the function before inputting to the homomorphic system, and phase unwrapping can be successfully carried out.

The use of exponential weighting can also improve the separability of the cepstra. If α is small enough so that all the zeroes of the reverberation sequence are scaled to lie within the unit circle, the sequence is converted to a minimum-phase sequence. Under this condition the reverberation sequence will contribute to the cepstrum only for positive times greater than or equal to the interval between the first two arrivals in the sequence,[2] and hence the separability of the cepstra will be improved. In contrast, the structure of the cepstrum of a mixed-phase sequence is complicated, difficult to predict, and frequently unstable.

Note that exponential weighting is different from the usual windowing procedure used in Fourier transformation, which is an approximation as far as convolution is concerned. In general, convolution is not conserved under multiplication by a windowing function: the product of a windowing function, $w(n)$, with the convolution of

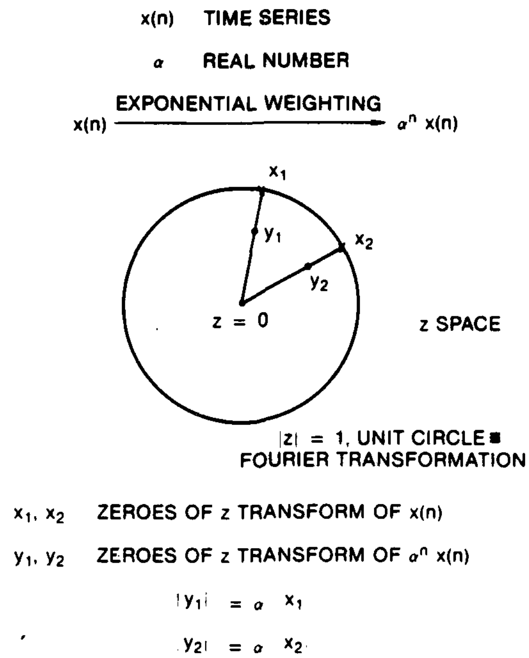


Figure 3. Removing zeroes in *z*-transform by exponential weighting

two functions, $f(n)$ and $g(n)$, is different from the convolution of $w(n)f(n)$ and $w(n)g(n)$:

$$w(n)\Big[f(n) * g(n)\Big] \neq \Big[w(n)f(n)\Big] * \Big[w(n)g(n)\Big]$$

Convolution will be conserved, however, if $w(n)$ is in the form α^n:

$$\alpha^n\Big[f(n) * g(n)\Big] = \Big[\alpha^n f(n)\Big] * \Big[\alpha^n g(n)\Big]$$

The output from the inverse homomorphic system is deweighted by the sequence $\{1/\alpha^n\}$. The modified homomorphic characteristic system including exponential weighting is illustrated in Figure 4.

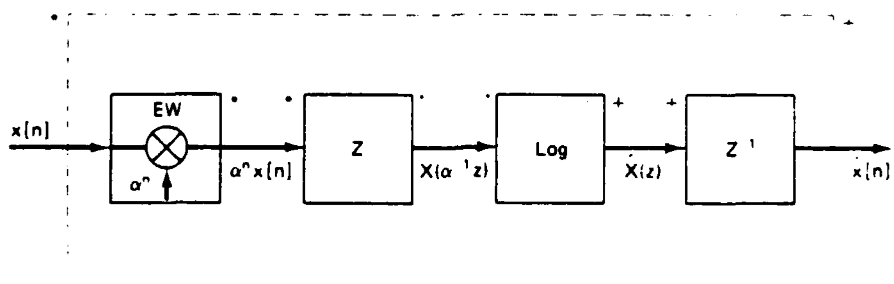


Figure 4. Modified homomorphic characteristic system including exponential weighting

2.5 Bandpass Mapping

If the signals one wishes to analyze have intrinsic bandpass characteristics, as, for example, the data taken with resonant transducers, homomorphic analysis cannot be accomplished by means of full-band homomorphic systems. In fact, the analysis cannot be performed on the unit circle of the z-plane, since then the logarithm would become unbounded in the frequency bands with zero energy. Neither can it be performed off the unit circle, since the z-transform of such signals does not converge anywhere on the z-plane, but on the unit circle. Thus the exponential weighting procedure used to remove the zeroes of a signal off the unit circle cannot be applied to the class of bandpass signals. Since real data always contain some out-of-band noise, and can only have a countable number of zeroes on the unit circle, sometimes it appears possible to employ the exponential weighting procedure to remove such zeroes from the unit circle. Such an approach, however, is inherently ill-conditioned, and often leads to erroneous results.

One way to solve this problem involves a restriction on the domain of the logarithmic mapping to encompass only the passband of the input. This approach may be conveniently formulated in terms of a frequency scaling operation that shifts and stretches the signal's passband to occupy the entire frequency domain, as illustrated in Figure 5. The result of this operation is then a full-band sequence, which may be analyzed using full-band homomorphic systems. The analyzed output from the homomorphic system undergoes the inverse bandpass mapping. The characteristic system for bandpass homomorphic systems is illustrated in Figure 6.

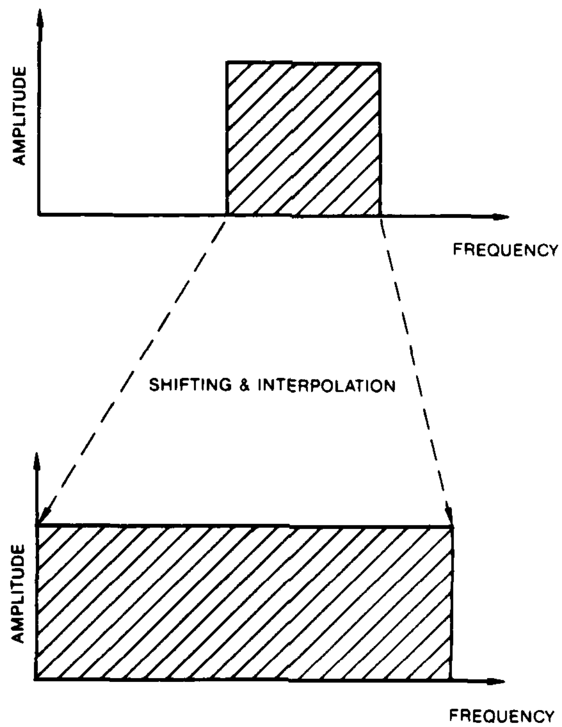


Figure 5. Bandpass mapping operation

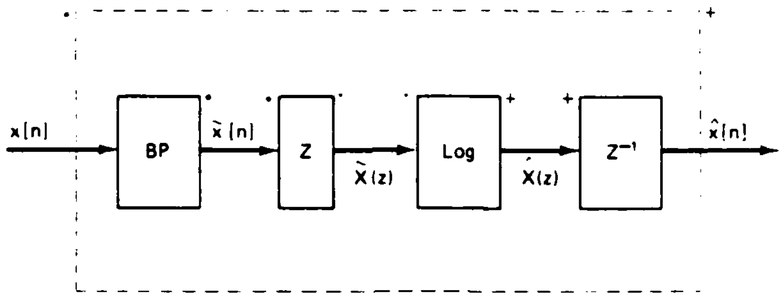


Figure 6. Characteristic system for bandpass homomorphic systems

Bear in mind that bandpass mapping is not a panacea, but rather a method to remove the effects of the out-of-band noise in phase unwrapping. It does not add any information in the frequency bands with zero energy and may not produce particularly satisfying results, especially if the system under analysis is sharply resonant, with a relatively small range of useful, information-carrying frequencies.

Section 3

EXPERIMENTAL APPARATUS

A VAX-based experimental system was assembled to provide a suitable testbed for the acoustic emission experiments carried out under this contract. The system, shown in schematic form in Figure 7, provided a great deal of flexibility in the acquisition, validation, and analysis of the acoustic emission data. Data acquisition, for example, was carried out under VAX control; this approach gave the experimenters access to VAX-based tools to monitor the newly acquired data and verify its integrity during the course of the experiments. In addition, the VAX system incorporated an interactive digital signal processing software system, which was used extensively in the analysis of the data; the system was useful not only in the exploratory stages while appropriate algorithms were being developed, but also during the production analysis phase when validated experimental data were being batch processed. Subsequent sections of this report will provide details on the hardware and software used in these experiments.

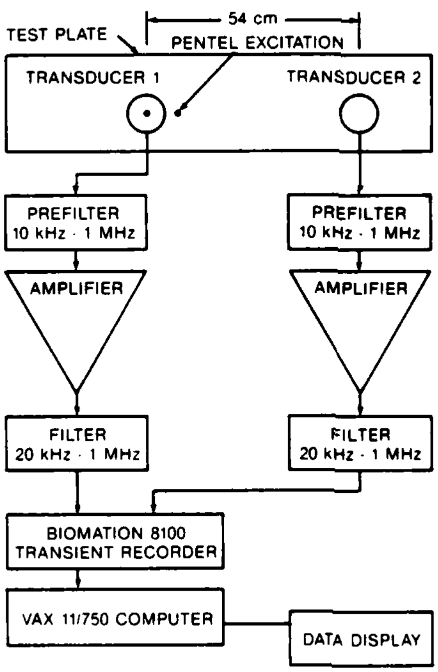


FIGURE 8 — BLOCK DIAGRAM OF THE DATA ACQUISITION SYSTEM.

Figure 7. Block diagram of the data acquisistion system

3.1 Hardware

The hardware used in the acoustic emission experiments will be described in the subsequent sections essentially in the same order as the signal flow through the system. Figure 8 shows the experimental setup; the thick test plate can clearly be seen, as can the transducers, the Biomation transient recorders, the analog instrumentation, and the VAX 11/750 control/analysis computer in the background.

Figure 8. The experimental setup, showing the thick test plate, the Biomation transient recorders, the analog instrumentation, and VAX 11/750 control/analysis computer

3.1.1 Specimens

A series of experimental specimens was prepared to provide a reasonable range of expected reverberation conditions; a key element in the design was the notion that it should be possible to increase gradually, and in a controlled manner, the physical complexity of the experiments, so that new conditions were added one at a time.

3.1.1.1 Workpieces. Two aluminum plates were selected to conduct the experiments. The first one was a relatively thick plate with dimensions of 27 × 5 × 2 in. The thick plate was chosen to allow clean near-field measurements without the confusing effects of reverberations from the back wall and other reflections. It functions in a near-field sense as an infinite half-sphere, and has known solutions for the expected AE due to a Pentel pencil lead break. The first plate is set up on a 2 in. thick foam pad to isolate

from ambient vibrations. The second plate had dimensions of 27 × 25 × 1/4 in. and was chosen to allow the investigation of the effects of plate bending vibrations and near-field resonance, as well as edge reflections. Only the thick plate was used for most of the experiments described in this report.

3.1.1.2 C-Block Stress Corrosion AE Sources. Notched aluminum C-blocks, under tensile stress and subject to a corrosive solution, were used to generate natural acoustic emissions. The notched C-blocks were clamped to the test plate and coupled with glycerine, so that good transmission of the generated AE pulses was achieved. A sodium chloride solution was subsequently used to induce stress corrosion cracking in the notched block. These sources provided reliable, natural AE signals, and could be easily controlled by small adjustments in the stress and/or salt solution so as to provide a wide range of natural AE signals, ranging from weak, individual pulses to nearly continuous trains of strong pulses.

3.1.2 Transducers

Both conventional resonant transducers (Physical Acoustics Corporation, Model PAC R15) as well as conical low-resonance wide-band transducers (Industrial Quality Inc., Gaithersburg, Md.) were used to detect the acoustic events. The resonant transducers have a band frequency response between 100 and 300 kHz. The conical transducers have a flat frequency response up to 1 MHz.

3.1.3 Analog Signal Conditioning Equipment

The signals from the transducers were first amplified using a Tektronix Model 502 amplifier with internal low-pass filter set at 1 MHz, or lower, depending on the nature of the sensing transducer. These signals were further processed using a Kronhite programmable filter (48 dB/octave/section) set up to low pass below 1 MHz. The output of the filter was then connected to the Biomation transient recorder system. A Nicolet digital Explorer scope was used for a visual display of the signals prior to transmitting them into the computer.

3.1.4 Transient Waveform Recorders

Biomation 8100 digital transient recorders were used to record and sample the acoustic emission signals. Using a single Biomation 8100, 2048 samples can be obtained in single-channel mode; in dual-channel mode, two parallel channels of 1024 samples can be obtained. The recorder operates at up to a maximum sampling rate of 100 MHz, if desired. For single-channel operation, a 2 MHz sampling rate results in the acquisition of nearly 1 ms of data. Longer records needed to verify the Phase I results were obtained by ganging together four of these systems in series. Using this arrangement, nearly 4 ms of continuous data could be acquired from a single AE event, if required. A VAX–Biomation interface board was designed and fabricated to provide both automated experiment control by the VAX, as well as a data path for the transmission of transducer signals from the Biomation recorder into the DEC VAX 11/750 computer for later postprocessing. The hardware and software for this data acquisition system were designed and put together during the duration of this contract specifically for these AE experiments.

3.1.5 Stress-Corrosion Event Monitor

A conventional AE transducer, in conjunction with a Dunegan/Endevco 920 Distribution Analyzer and a Tektronix 604 Monitor, was used to monitor the rate of arrival and the intensity of the stress-corrosion events. This information allowed us to categorize easily the state of the C-block stress corrosion AE source, as well as providing a useful reference standard against which to measure the new analysis methods under evaluation.

3.2 Software

A variety of software modules were employed or developed to serve the needs of the experimental effort. Software modules were developed or used for data acquisition, data validation, algorithm identification, homomorphic data analysis, feature extraction, and pattern recognition. Much of the data analysis was carried out using an interactive, interpretive, high-level digital signal processing language available from Signal Technology Inc. (Santa Barbara, Calif.) called ILS. "Recipes," consisting of DEC DCL command procedures interpretable by ILS, are provided for all the analysis carried out using ILS. In some cases, where ILS did not provide appropriate functional modules, new ILS procedures were developed to provide the needed functionality. In either case, the modules used will be functionally described below; source code, and documentation for those developed under this contract, can be found in the appendices.

3.2.1 Data Acquisition

Software was developed to interface the Biomation hardware with the VAX 11/750 computer and to provide some measure of validation for the experimental data. Additional software was developed to prepare the input data in a form compatible with the high-level signal processing language (ILS) which contains a variety of software modules for data analysis.

3.2.1.1 VAX–Biomation Interface. A data acquisition/control software system was put together to acquire data with, and control up to, four Biomation 8100 transient recorders with a VAX 11/750. The Biomation interface software is MACRO-based and controls a DR-11C parallel interface board. The interface permits VAX-based software control of the Biomation recorders. All of the Biomation front panel switches and controls can be set via software commands originating in the VAX. In use, the Biomation transient recorders are operated in parallel, with one of the recorders providing triggering synchronization signals for the others. The trigger delays were set so that the captured data records would overlap slightly. The captured data records were processed by a VAX FORTRAN program that carried out correlation calculations on the overlapped regions to verify that the record segments were correctly synchronized, making appropriate corrections if required. In addition, the overlapped regions were used to adjust the gain and offset on each record to compensate for record-to-record variations. Finally, graphics software permitted the experimenter to view the captured records, or any subset of them, prior to proceeding. Complete documentation on the system, including functional descriptions, schematics, and software listings can be found in Appendix A.

3.2.2 Analysis

The basic approach in developing analysis software was to carry out as much of the analysis as possible directly in the ILS (see Appendix B) digital signal processing language; ILS is an expressive language, licensed by Signal Technology Incorporated, and provides a flexible ensemble of digital signal processing primitives built into its structure. In addition, ILS provides many kinds of generalized support functions, including data editing capability, simulation software, graphics software, and extensive pattern recognition software. ILS graphics capability, for example, supports graphics primitives for either interactive or hardcopy display, along with the associated support functionality.

Many complex functions can be constructed by stringing together primitive functional modules into DCL command procedure recipes. These recipes are executable either on a stand-alone basis, or as new primitive modules in more complex primitives. All the recipes used in analyzing the data in this report are included, where appropriate, with the figures in Section 4 displaying processed data. Finally, it is relatively easy to add entirely new modules to the ILS system.

Perhaps the most important feature of ILS is that it is intended to be used by nonprogrammers; thus it allows a researcher familiar with digital signal processing techniques to explore new techniques very quickly. This approach allowed considerable flexibility, and proved to be a very effective way to analyze and process quickly the acoustic emission data acquired under this contract.

3.2.2.1 Homomorphic Analysis. A software package was developed under Phase I at Rensselaer Polytechnic Institute to carry out simulation studies on the application of homomorphic analysis to acoustic emission signals. This package was based on the routines published in the IEEE volume *Programs for Digital Signal Processing.*[5] After some changes and modifications, the package was installed in the VAX computer system and integrated with the ILS digital signal processing package as the XCP module. Besides altering the code to conform to the ILS functional conventions, two basic changes were made to the existing code. The first change consisted of adding variable exponent exponential weighting capability to convert the acquired waveforms to minimum phase sequences, which are analytically more stable than mixed phase sequences. (See Section 2.4 for a description of the theory behind exponential weighting.) The second change involved adding bandpass mapping, necessary to deal with the signals produced by narrow-band resonant transducers. (See Section 2.5 for a description of the theory behind bandpass mapping.) The source code and documentation for the XCP ILS module can be found in Appendix B.

3.2.2.2 Feature Extraction and Pattern Recognition. A feature extraction and pattern recognition package was put together to investigate the separability of the different groups of waveforms. The package was based generally on the pattern recognition software available in ILS. Since the ILS pattern recognition modules were originally designed for speech and speaker recognition, minor changes had to be made to some of the modules to make them suitable for this application. The process followed in using the pattern recognition package to analyze acoustic emission waveform data was the traditional training/test set approach. A set of waveforms is analyzed and used to train the system; subsequent sets are tested against the training set for statistical compatibility. Summarizing the process, each of the waveforms in the group is Fourier transformed to extract a number of Fourier components. These Fourier components

are then processed to yield the mean waveform, the covariance matrix, the Fischer linear discrimination matrix, and the eigenvalues and eigenvectors for each group of waveforms. Using these intermediate results, principal component analysis is carried out on sets of waveforms. In essence, principal component analysis projects the groups of waveforms onto a discriminant plane with maximum separation between the groups. A scatter plot is then made using the first two principal components, and the statistical properties of each of the groups can be computed. Finally, the mean separation between groups can be expressed in statistical terms and used to characterize the probability that a given test value is in fact a member of the training set. A complete description of the software modifications to the ILS modules can be found in Appendix B.

Section 4

EXPERIMENTAL RESULTS

The block diagram of the experimental layout, hardware, and associated electronics was shown in Figure 7 of the previous section.

A problem encountered very early in the analysis was that our available methods for recording and for carrying out analog-to-digital conversion of the data did not provide sufficiently long record lengths for our analysis. Initially, the data acquisition system utilized a single Biomation 8100 interface providing a pair of 1024-point-long records. For sampling rates of 10 MHz, this system provided records approximately 0.1 ms in duration, insufficiently long for reverberations to decay in the experimental fixtures we employed. Accordingly, several different hardware approaches for recording and analog-to-digital conversion were explored to try to identify a more suitable means of acquiring the large experimental data sets needed to confirm the results of Phase I. The first approach explored was to reduce the sampling frequency to 2 MHz and to use the Biomation recorder in single-channel mode, so that a single 2048-point record is obtained instead of a pair of 1024-point records. This technique yields a record duration of roughly 1 ms, which—although still too short to record the entire ringdown train—provided enough data to begin analysis.

Later the record length was increased fourfold to 8096 points by connecting four Biomation recorders in series, as described above in Section 3.1.4. At a sampling rate of 2 MHz, this arrangement gives a record length of about 4 ms, which was adequate for subsequent analysis.

The command procedures for generating the ILS plots shown in Figures 9 and following are included in the illustrations. The DSP command at the end of most of the instruction sets displays the result in sampled data form.

4.1 Calibration

The breakage of Pentel pencil leads was employed as a calibration source. In the initial phase of the experiments, the 0.5 mm HB leads were broken manually.

Two typical lead-breaking waveforms are shown in Figures 9 and 10. They are two separate events taken with two of the wide-band, NBS-style conical transducers spaced about 54 cm apart on the 27 × 5 × 2 in. aluminum block. Pentel lead breaks were generated relatively close to transducer 1 at the location indicated. Signals from transducer 1 triggered the Biomation to accept data. The Biomation was set to the pretrigger mode. The data were low-passed at 1 MHz with the Kronhite filter, sampled at 2 MHz, for a 2048-point record. The waveform shown in Figure 9 is the output from transducer 1, while the waveform shown in Figure 10 is from transducer 2.

The time difference between the two pulses in Figures 9 and 10 is about 200 μs, which is reasonably close to the value of 176 μs calculated from the geometry of the setup, assuming the shear wave speed in aluminum to be 3.1 km/s. The waveforms exhibit low-frequency ringdown due to the reflections in the aluminum block.

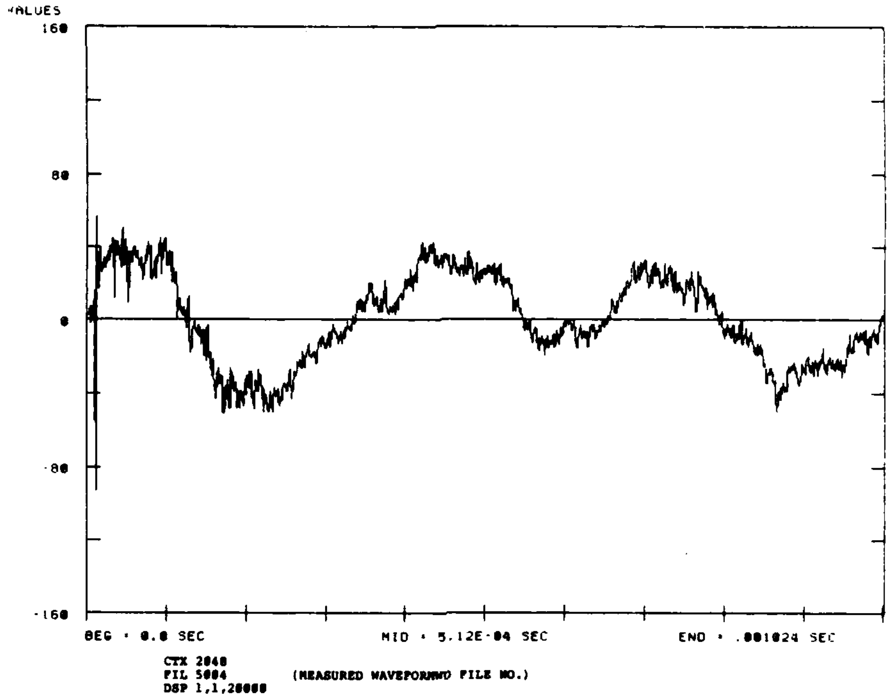


Figure 9. A typical lead-break waveform recorded by an NBS conical transducer

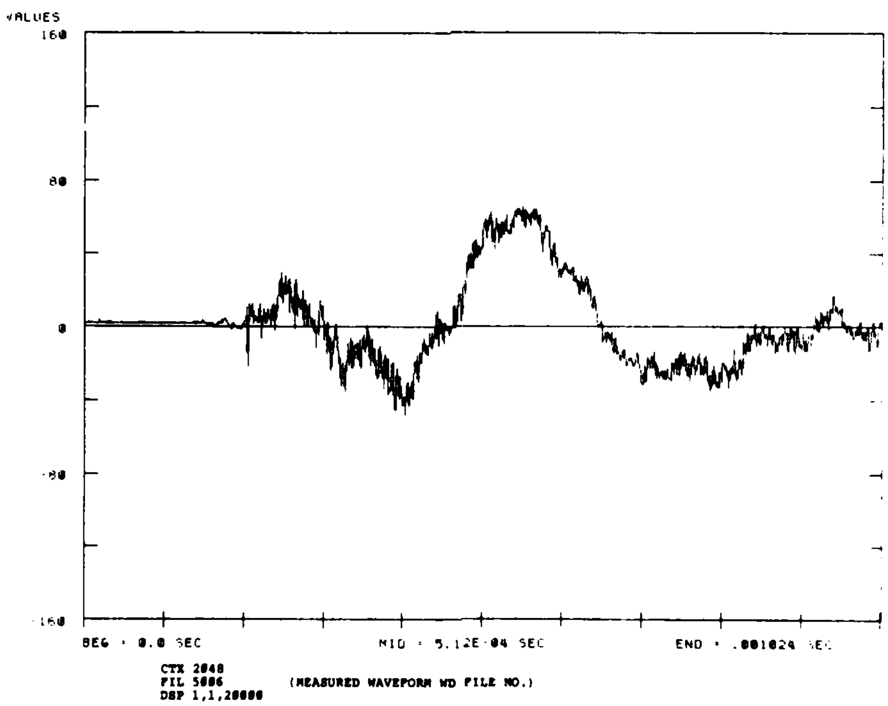


Figure 10. Another lead-break waveform recorded by an NBS conical transducer located 54 cm from the one used in Figure 9

Homomorphic deconvolution was performed on the waveform in Figure 10. The cepstrum of the waveform is shown in Figure 11. The result of passing the cepstrum through the low-time window $w(t)$:

$$w(t) = \begin{cases} 1 & -9.5\ \mu s < t < 9.5\ \mu s \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

and then inverse transforming is shown in Figure 12. An impulse type of function is recovered, but the result is not very clean: the ringing in the background is quite serious.

A large improvement was brought about by employing the technique of exponential weighting to minimize the effects of reverberation. The result of applying exponential weighting to the waveform in Equation 10 with $\alpha = 0.9955$ and using the low-time window given by Equation 5 is shown in Figure 13. The recovered signal is much sharper and cleaner, and in fact compares favorably with the theoretical Pekaris solution, which is shown in Figure 14.

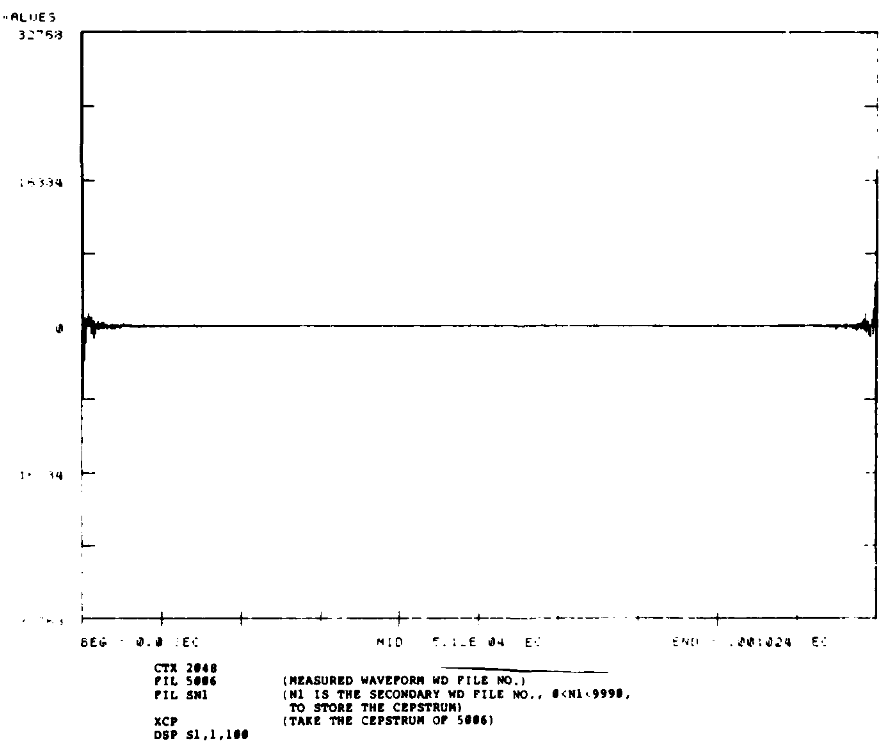


Figure 11. The cepstrum of the waveform in Figure 10

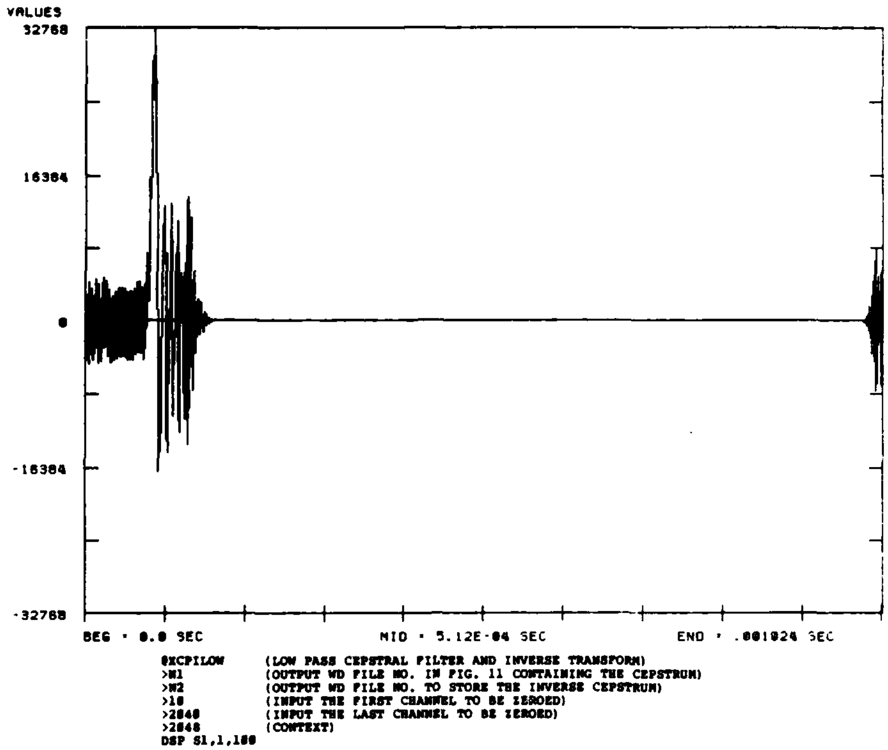


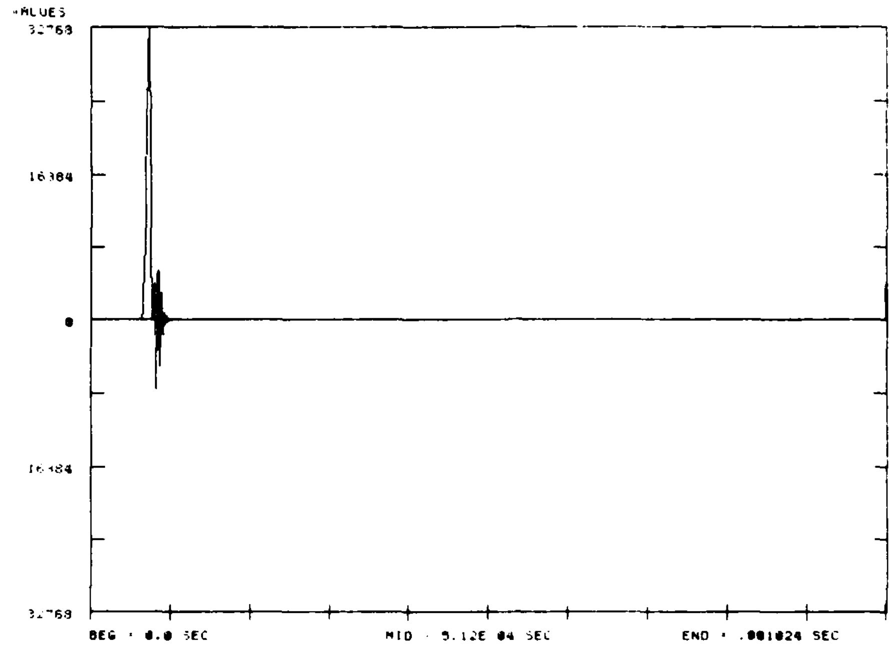
Figure 12. The result in the time domain of low-time pass filtering the cepstrum of the waveform in Figure 10

4.2 Effect of Record Length

Another problem encountered was that the ringdown response of the medium could last substantially longer than the maximum record length, for as long, in fact, as several seconds under some circumstances. The record length of a waveform determines the fraction of ringdown captured, and hence the amount of leakage error in the Fourier transformation operation in homomorphic analysis. One would certainly expect the results to improve substantially when the record length is increased. As we shall see, however, if the effect of the record length is not significant enough, the improvement brought about by longer record length could easily be masked by other sources of errors.

The effect of the record length is demonstrated in the results shown in the sequence of Figures 15–26. Figures 15–18 are four lead-breaking waveforms captured with a conical transducer, the source being located at distances 45, 45, 30, and 15 cm, respectively, from the transducer. The signals were low-pass filtered at 1 MHz, sampled at 2 MHz, and are 8096 points in length. Each of these waveforms was transformed to the cepstral domain, then low-time filtered with the $w(t)$:

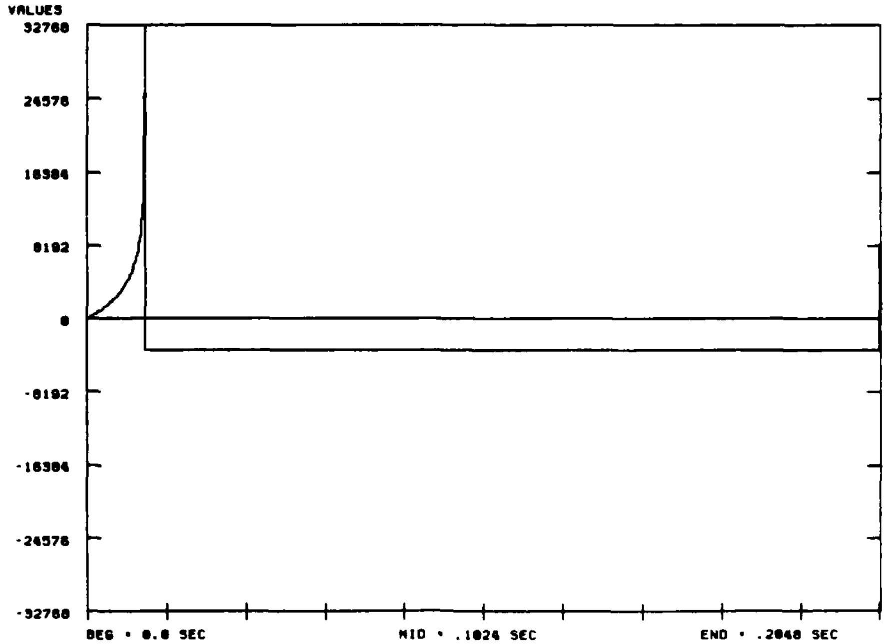
$$w(t) = \begin{cases} 1 & -4.5\ \mu s < t < 4.5\ \mu s \\ 0 & \text{otherwise} \end{cases} \tag{6}$$




```
CTX 2048
FIL 5006
FIL SN1       (N1 IS THE SECONDARY WD FILE NO., 0<N1<9990,TO STORE
               THE CEPSTRUM)
XCP,,1        (TAKE THE CEPSTRUM WITH EXPONENTIAL WEIGHTING)
>0.9955       (INPUT THE WEIGHTING PARAMETER ALPHA)
@XCPILOW      (SEE DESCRIPTION IN FIG. 12)
>N1
>N2          (N2 IS THE SECONDARY WD FILE NO., 0<N2<9990,TO STORE
              THE INVERSE CEPSTRUM )
>10
>2048
>2048
DSP S1,1,100
```

Figure 13. The improved result obtained through the use of exponential weighting; $\alpha = 0.9955$

and finally inverse-transformed to the time domain. The value of α used in exponential weighting is 0.9985. Figures 19 through 22 show the results obtained in this way using all of the 8096 samples of the input waveforms in Figures 15 through 18. In comparison, the results obtained using only the first 2048 samples of the input waveforms are shown in Figures 23 through 26. Identical values of α and $w(t)$ are used in both cases. In comparing the two sets of results, although Figure 19 is much cleaner than Figure 23, the improvement caused by the longer record length is less obvious in the other three cases. From these results it may be concluded that the leakage error in Fourier transformation in going from 4.096 ms (corresponding to 8192 points at 2 MHz sampling rate) to 1.024 ms is not significant compared to the other sources of errors in the experiments and analysis. The lack of improvement with longer record length may also be due partially to some transient instabilities in the gain and noise characteristics of the four Biomation recorders. At any rate, the record length of 1.024 ms provided by one Biomation recorder at a sampling rate of 2 MHz appeared to be long enough for our purpose. The waveforms used in some of our later analyses were 1.024 ms in length.




```
CTX 2048
FIL 1001
FIL S1002
FIL SB1003
FIL SC1004
@TCP                (BUILD THE PROGRAM TO GENERATE THE SIMULATED
                     WAVEFORM)
TCP F               (RUN THE PROGRAM)
FIL 1003
DSP 1,1,100
```

Figure 14. The theoretical seismic surface pulse according to Pekaris

4.3 Dependence on the Width of the Cepstral Filter

The ideal width of the cepstral filter used to separate out the signal impulses from the recorded waveforms was determined empirically. The results indicated that in most cases a filter centered around the origin in the cepstral domain with a width of about 4.5 μs worked best. Two typical sets of results are illustrated in Figures 27–32. Figures 27–29 show the results of low-passing the cepstrum of the recorded waveform in Figure 10 using $\alpha = 0.999$ in exponential weighting and cepstral filters of widths ± 2.5, ± 4.5, and ± 9.5 μs, respectively. Figures 30–32 are the results of applying the same procedure to the waveform in Figure 15 using $\alpha = 0.9985$ in exponential weighting. In both cases, the cepstral filter with a width of ± 4.5 μs yielded the best results.

4.4 Dependence on Alpha

In the homomorphic analysis of single-event waveforms, exponential weighting converts the original time sequences $h_p(n)$ and $h_r(n)$ to the new ones $\alpha^n h_p(n)$ and $\alpha^n h_r(n)$. The value of α determines the distributions of the cepstra of the two exponentially weighted time sequences, and hence the results of the homomorphic deconvolution depend on the value of α used in the exponential weighting. To study the extent of this dependence, homomorphic analysis was performed on the waveform in Figure 15 with the window $w(t)$ in Equation 6 but with different values of α in the

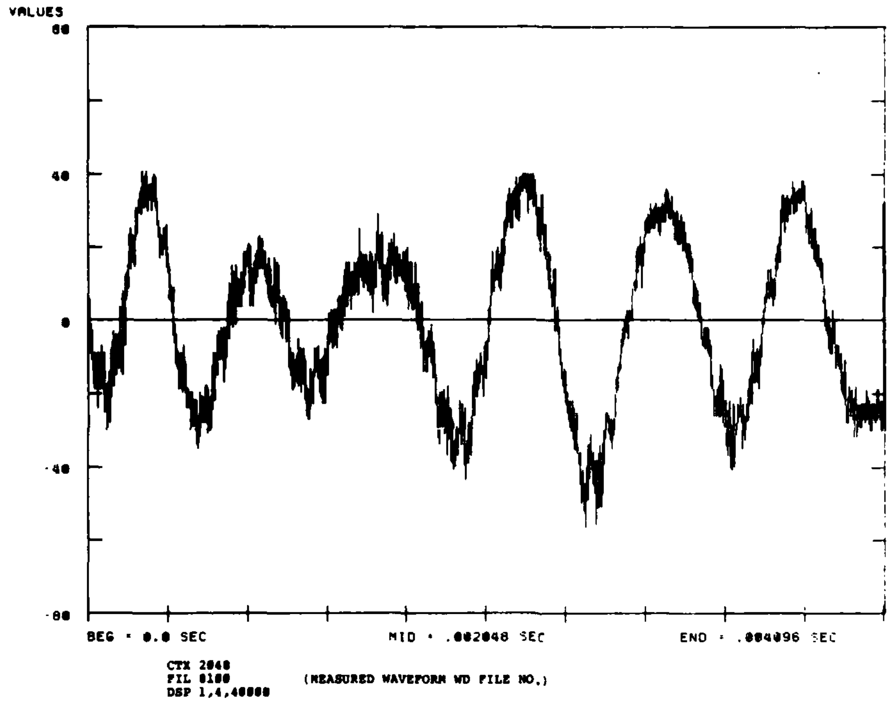


Figure 15. A lead-break waveform recorded by an NBS conical transducer located 45 cm from the source

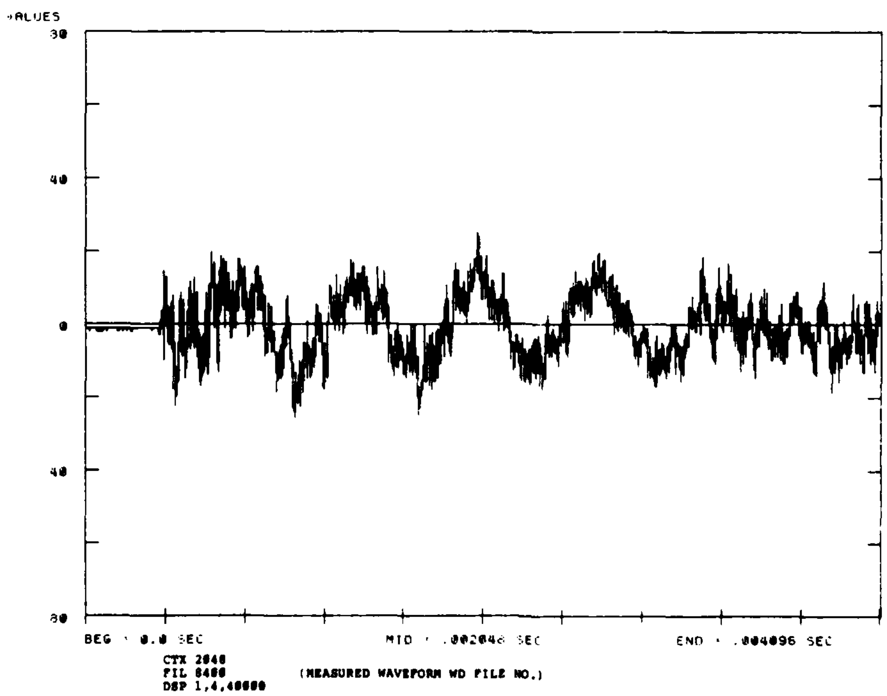


Figure 16. A lead-break waveform recorded by an NBS conical transducer located 45 cm from the source

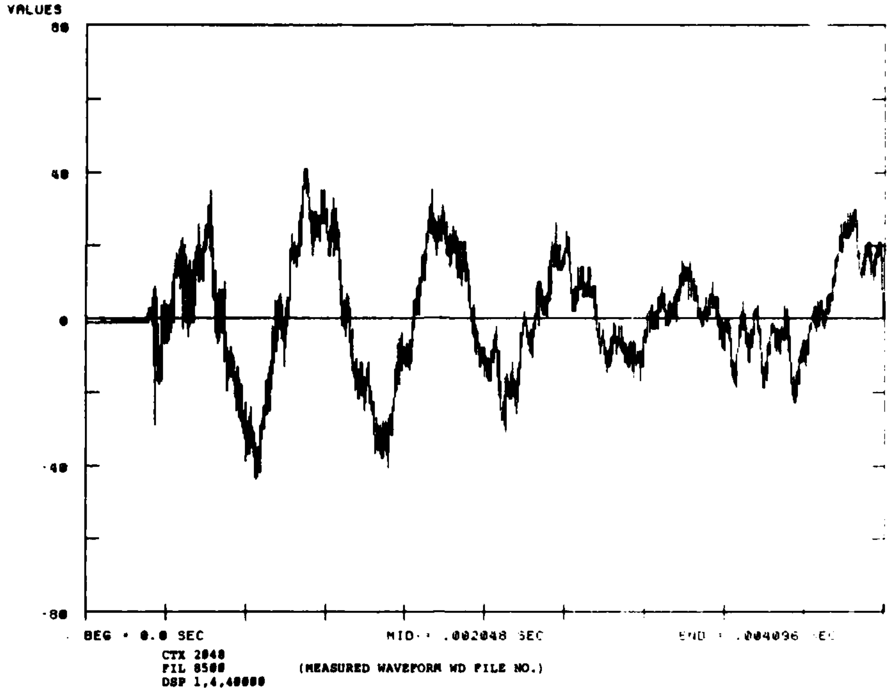


Figure 17. A lead-break waveform recorded by an NBS conical transducer located 30 cm from the source

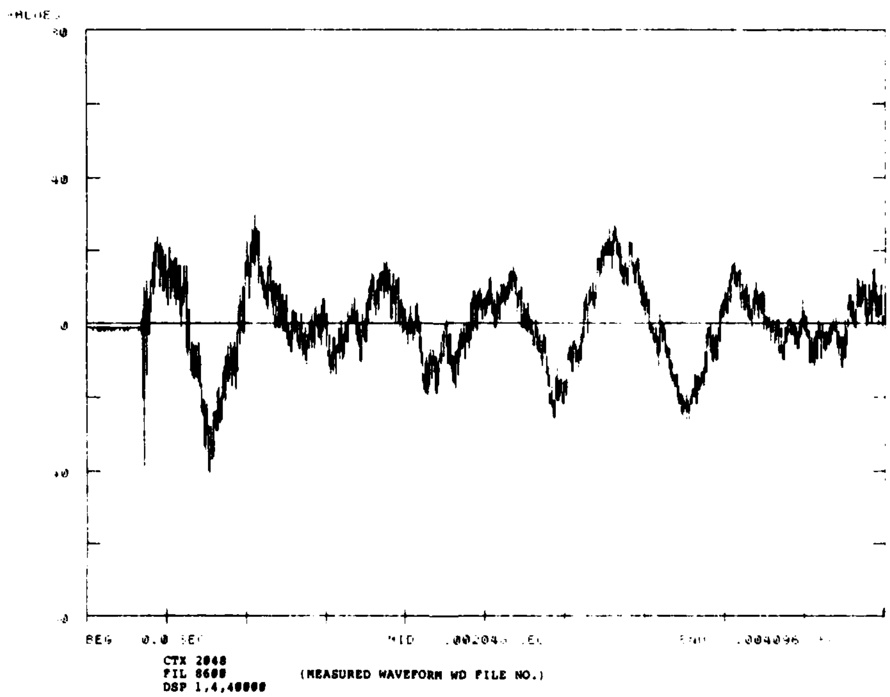
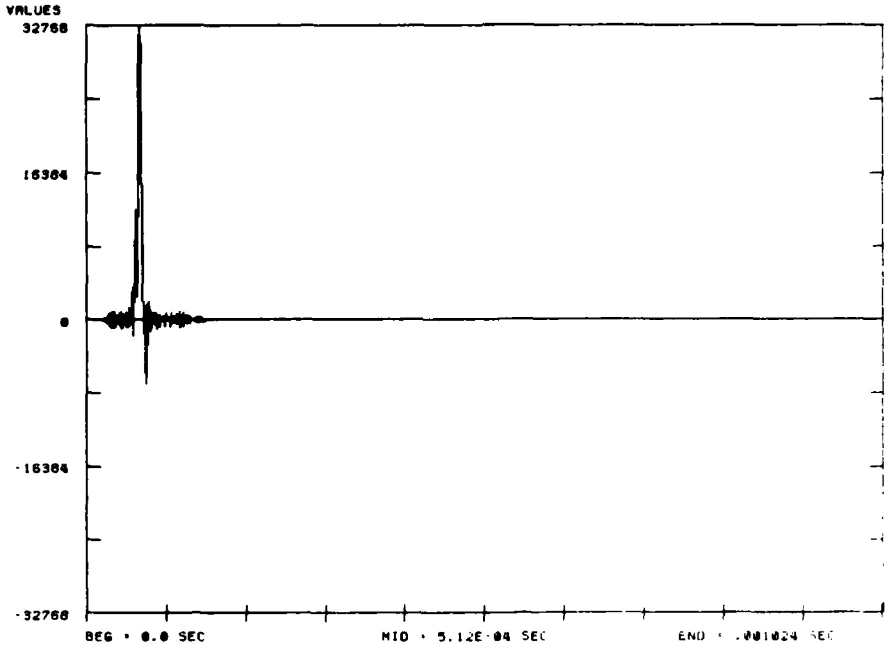


Figure 18. A lead-break waveform recorded by an NBS conical transducer located 15 cm from the source




```
CTX 8192
FIL 8100        (MEASURED WAVEFORM WD FILE NO.)
FIL SN1         (N1 IS THE SECONDARY WD FILE NO., 0<N1<9990,
                 TO STORE THE CEPSTRUM)
XCP,,1          (SEE THE DESCRIPTION IN FIG. 13)
>0.9985
@XCPILOW        (SEE THE DESCRIPTION IN FIG. 12)
>N1
>N2             (N2 IS THE SECONDARY WD FILE NO., 0<N2<9990,
                 TO STORE THE INVERSE CEPSTRUM)
>10
>8104
>8192
CTX 2048
DSP S1,1,100
```

Figure 19. The result of low-time pass filtering the cepstrum of the waveform in Figure 15, using all of the 8096 samples as input; $\alpha = 0.9985$

exponential weighting. The results corresponding to $\alpha = 0.9985$, 0.998, and 0.9975 are shown in Figures 33, 34, and 35, respectively. The results indicate that even a slight change in the value of α affects the results significantly.

4.5 Fourier Deconvolution

Initial efforts to use the transfer function obtained in the homomorphic deconvolution of one Pentel lead-breaking waveform to deconvolve other Pentel lead-breaking waveforms were met with little success. To understand the difficulties, the repeatability of the manually performed lead-breaking waveforms was examined in detail. It was found that the lead-breaking signals depend critically on (1) the precise angle the lead makes with the surface and (2) possible secondary Pentel tip impact after the lead break.

These findings were illustrated in Figures 36–38, which are manually performed lead-breaking waveforms captured with a conical transducer. Figures 36 and 37 illustrate the dependence of the signal on the angle between the lead and the surface. The waveform in Figure 36 was generated with an angle of about 65°, and the one in Figure 37 with an angle of about 40°. The waveforms are markedly different. It appears

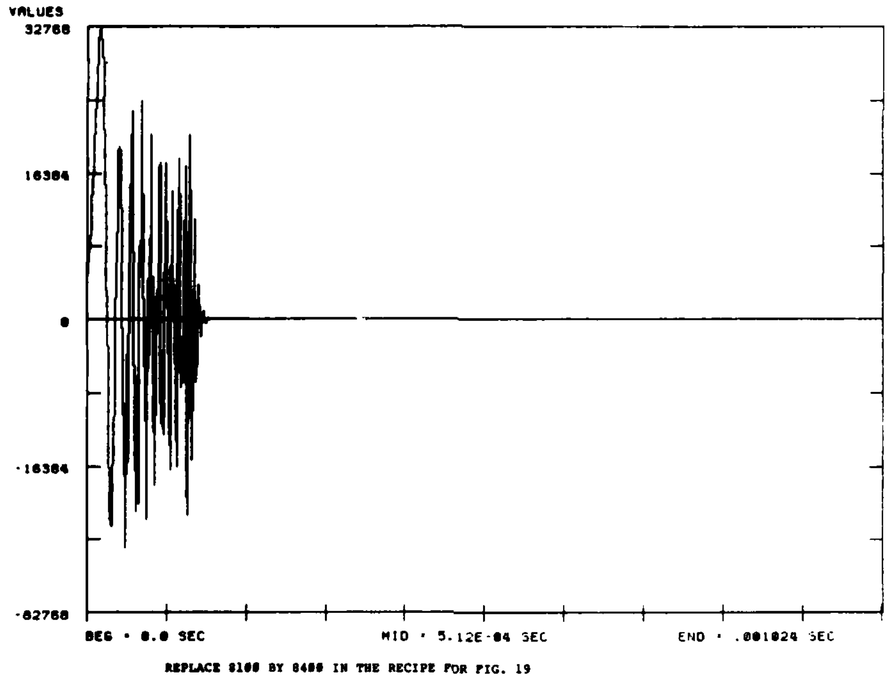


Figure 20. The result of low-time pass filtering the cepstrum of the waveform in Figure 16, using all of the 8096 samples as input; $\alpha = 0.9985$

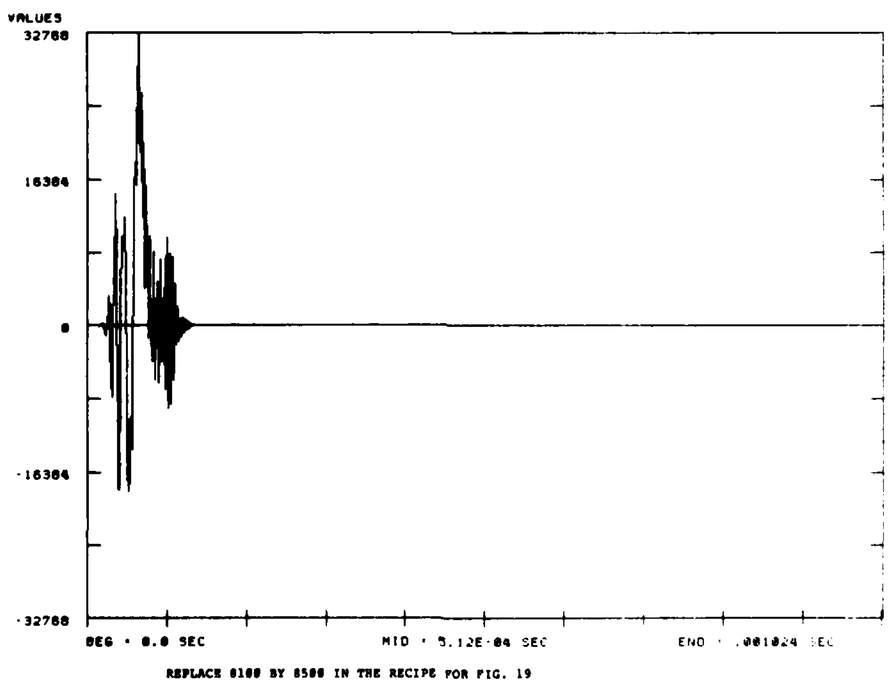


Figure 21. The result of low-time pass filtering the cepstrum of the waveform in Figure 17, using all of the 8096 samples as input; $\alpha = 0.9985$

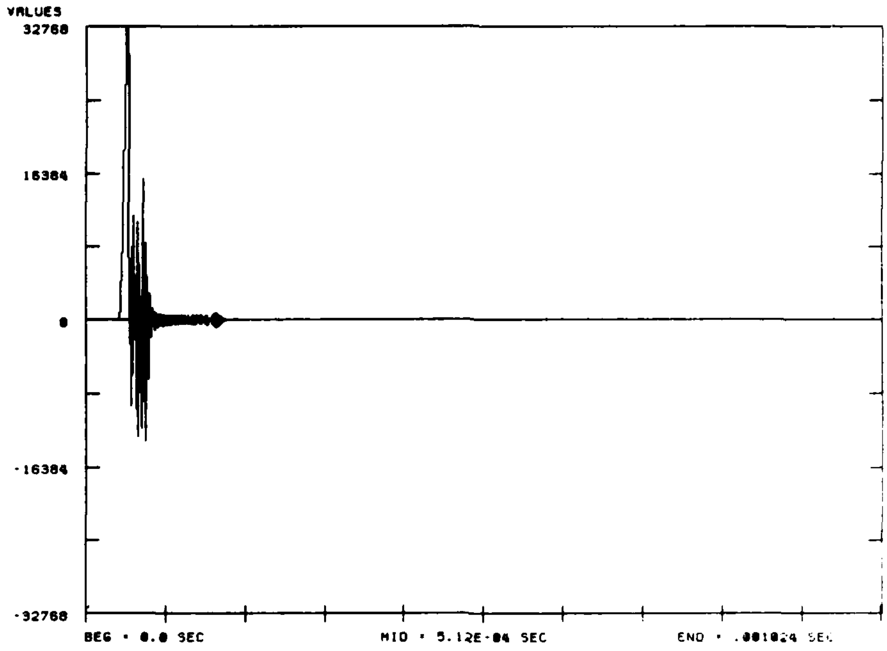


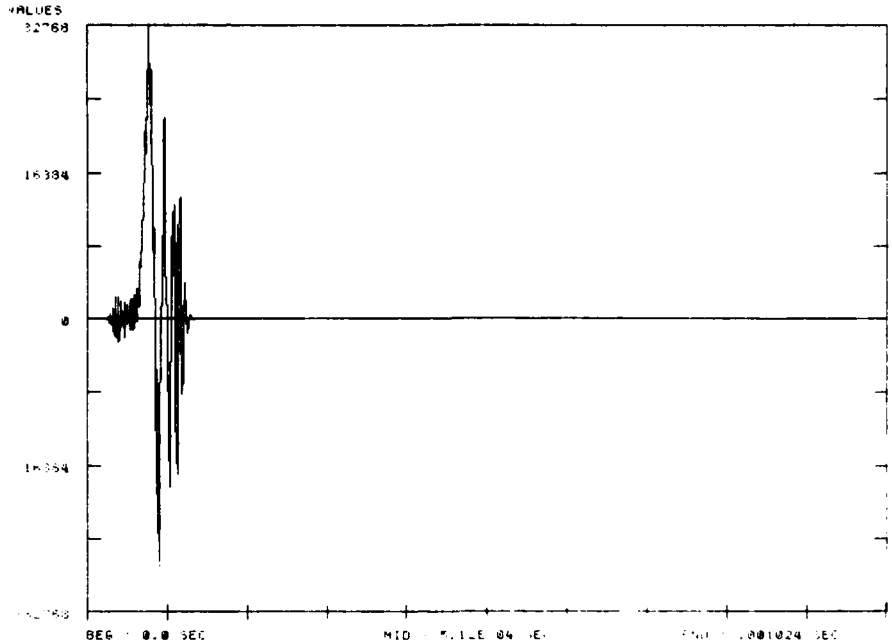
Figure 22. The result of low-time pass filtering the cepstrum of the waveform in Figure 18, using all of the 8096 samples as input; $\alpha = 0.9985$

that the angle determines the amount of shear and transverse waves generated in each event. A shallow-angle (<45°) event contains relatively more low-frequency components and is relatively featureless. On the other hand, a steep angle (~60°–70°) produces strongly featured signals with much modulation. A Pentel lead break with secondary Pentel tip impact at large angle is shown in Figure 38. The secondary impact generated additional AE activity in the midregion of the waveform.

In an effort to produce repeatable lead-breaking signals, a fixture for breaking leads in a well-controlled manner was acquired. The fixture allowed precise control over the angle of the lead during breakage and effectively eliminated problems with periodic secondary tip contact. Further studies on the feasibility of Fourier deconvolution were carried out using waveforms generated with the help of the fixture. It was found that the waveforms were highly repeatable. The repeatability is demonstrated in Figures 39 and 40, where 200 points of two such waveforms are shown near the initial impulses. These waveforms were recorded by a conical transducer and sampled at 2 MHz. Figures 41 and 42 show another 200 points in the later part of the two waveforms. Minor differences between the two waveforms are visible after a considerable delay, but the overall similarity is still very impressive.

Fourier deconvolution using these mechanically generated waveforms yielded encouraging results. Figure 43 shows the impulse response obtained by the following procedure:

- Averaging 33 of such waveforms (Pentel lead breaking by a mechanical device recorded by a conical transducer), each of the waveforms being 1024 points in length




```
CTX 2048
FIL 8100              (MEASURED WAVEFORM WD FILE NO.)
FIL SO
>CZN1,,1,2048         (N1 IS THE SECONDARY WD FILE NO., 0<N1<9990,
                       TO STORE THE FIRST 2048 CHANNELS OF FILE 8100)

MDF S0,63,-32000
TRF H
TRF 1,1,1,100         (COPY THE FIRST 2048 CHANNELS OF FILE 8100
                       TO FILE N1)
FIL N1
FIL SN2               (N2 IS THE SECONDARY WD FILE NO., 0<N2<9990,
                       TO STORE THE CEPSTRUM)
XCP,,1                (SEE THE DESCRIPTION IN FIG. 13)
>0.9985
@XCPILOW              (SEE THE DESCRIPTION IN FIG. 12)
>N2
>N3                   (N3 IS THE SECONDARY WD FILE NO., 0<N3<9990,
                       TO STORE THE INVERSE CEPSTRUM)
>10
>2040
>2048
DSP S1,1,100
```

Figure 23. The result of low-time pass filtering the cepstrum of the waveform in Figure 15, using only the first 2048 samples as input; $\alpha = 0.9985$

- Calculating the cepstrum of the averaged waveform, $\alpha = 0.999$
- High-pass filtering the cepstrum, i.e., setting the low-time portion of the cepstrum to zero for $|t| < 4.5\ \mu s$
- Inverse-transforming the cepstrum to the time domain

The result of deconvolving one of the 33 waveforms used in averaging, the one shown in Figure 44, by the impulse response in Figure 43 is illustrated in Figure 45. The result was improved by removing the high-frequency noise with the elliptical frequency filter shown in Figure 46 with -60 dB cut-off at 700 kHz; the filtered result is show in Figure 47.

The deconvolution results shown in Figures 45 and 47 were obtained under close to ideal conditions:

- Obtaining highly repeatable waveforms by breaking Pentel leads with a mechanical device

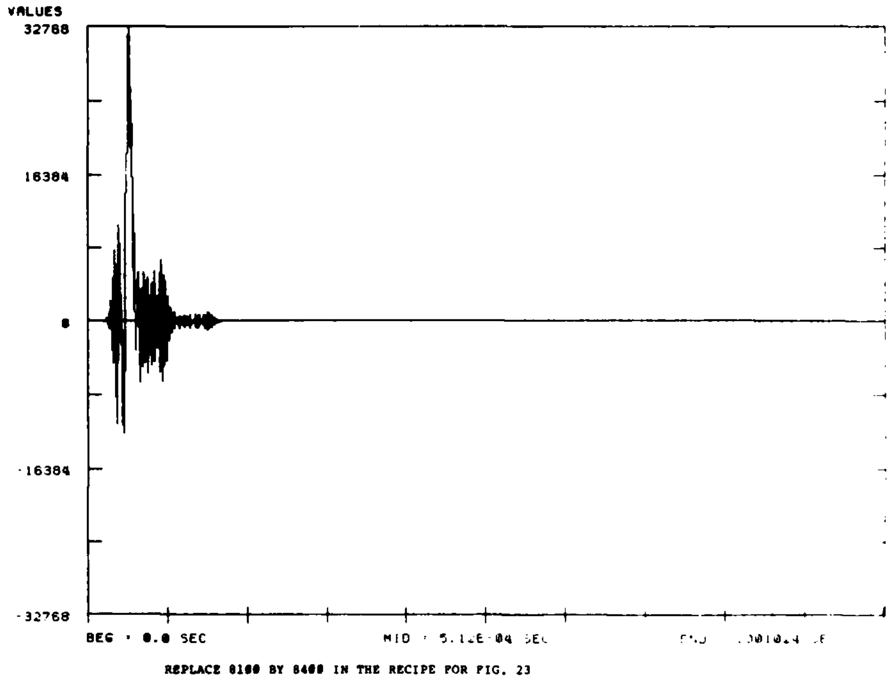


Figure 24. The result of low-time pass filtering the cepstrum of the waveform in Figure 16, using only the first 2048 samples as input; $\alpha = 0.9985$

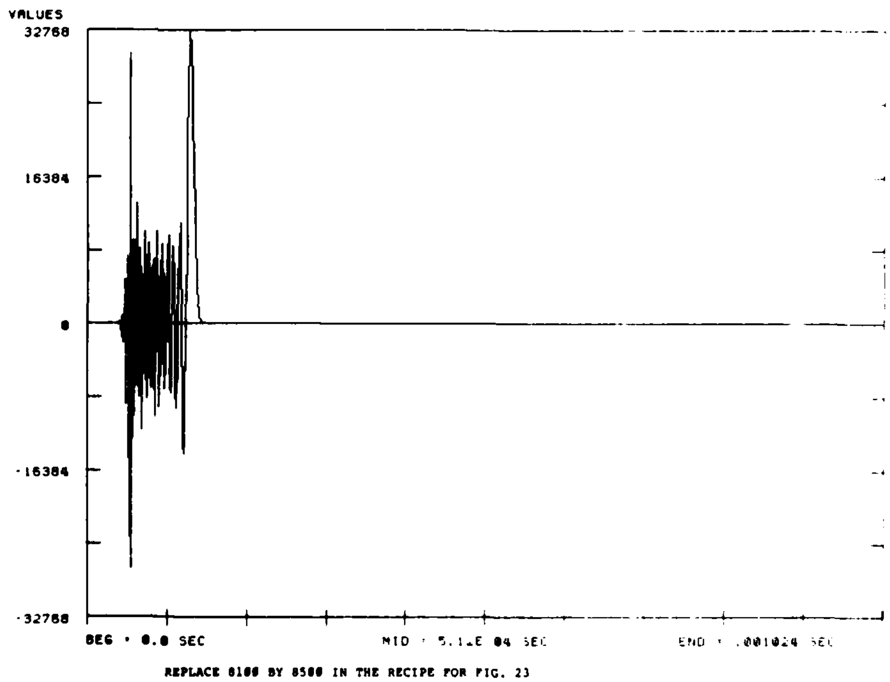


Figure 25. The result of low-time pass filtering the cepstrum of the waveform in Figure 17, using only the first 2048 samples as input; $\alpha = 0.9985$

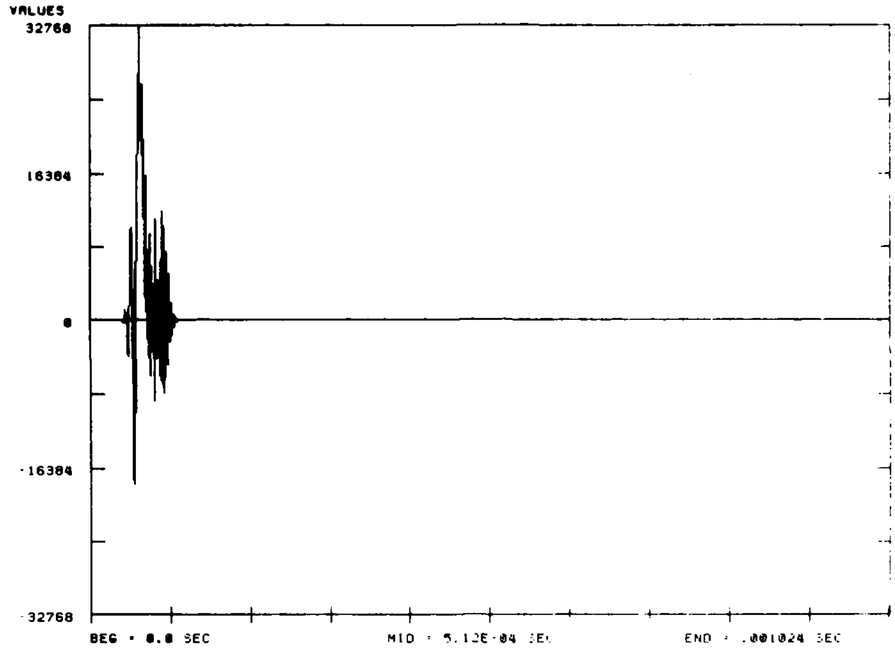


Figure 26. The result of low-time pass filtering the cepstrum of the waveform in Figure 18, using only the first 2048 samples as input; $\alpha = 0.9985$

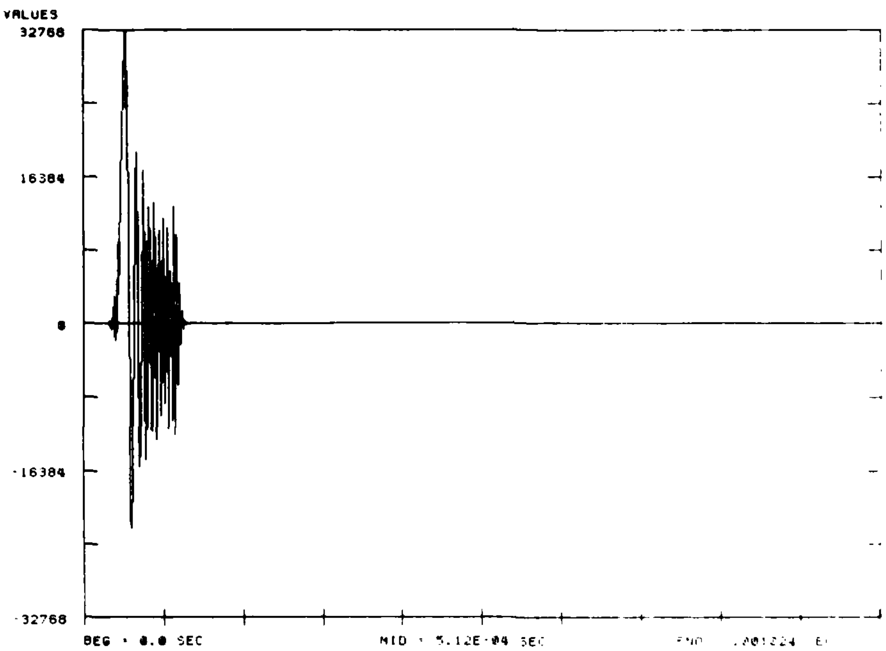
- Recording the waveforms with a wide-band conical transducer
- Using a relatively short record of 1024 points, where the similarity between waveforms is extremely good

In Figures 45 and 47, the presence of the impulse is obvious, yet the quality of the pulse shape does not appear to be good enough for identification purposes, even after low-pass filtering to remove high-frequency noise. These results indicate that Fourier deconvolution is too sensitive to the minor differences in the recorded waveforms to be of any value for event identification in practical situations.

4.6 Effects of a Limited-Frequency Band

Realistic acoustic emission events were generated using aluminum C-blocks subject to stress and a salt solution. It was found that the amplitude of stress-corrosion events generated in this way was about 60 dB lower than the amplitude from Pentel lead breaks; this finding can be seen clearly in the amplitude plot shown in Figure 48. Unfortunately, since the sensitivities of the conical transducers are much lower than those of the resonant transducers, the stress-corrosion signals could not be detected by the conical transducers at all. For this reason, resonance transducers were used for detection.

The data taken with the resonance transducers are band-limited between 100 kHz and 300 kHz. The power spectrum of one of the stress-corrosion waveforms captured with a resonance transducer is shown in Figure 49. As mentioned in Section 2.5, such data require bandpass mapping to suppress the effects of noise propagation in the




```
CTX 2048
FIL 5006            (MEASURED WAVEFORM WD FILE NO.)
FIL SN1             (N1 IS  THE SECONDARY WD FILE NO., 0<N1<9990,
                     TO STORE THE CEPSTRUM)
XCP,,1              (SEE THE DESCRIPTION IN FIG. 13)
>0.999
@XCPILOW            (SEE THE DESCRIPTION IN FIG. 12)
>N1
>N2                 (N2 IS THE SECONDARY FILE NO., 0<N2<9990,
                     TO STORE THE INVERSE CEPSTRUM)
>6
>2044
>2048
DSP S1,1,100
```

Figure 27. The result of low-time pass filtering the cepstrum of the waveform in Figure 10, $\alpha = 0.999$, width of cepstral filter = $\pm 2.5\ \mu s$

unoccupied frequency band in the process of phase unwrapping. Figure 50 shows the average of 31 stress-corrosion waveforms with their initial impulses lined up at the same location, as mentioned in Section 2.3.2. The waveforms were captured with a resonance transducer, were sampled at 2 MHz, and were 2048 points in length. The result of low-pass filtering the cepstrum of the averaged waveform without bandpass mapping is shown in Figure 51. The value of α used in exponential weighting is 0.999, and the low-time cepstral filter $w(t)$ is given by

$$w(t) = \begin{cases} 1 & -4.5\ \mu s < n < 4.5\ \mu s \\ 0 & \text{otherwise} \end{cases}$$

Figure 52 shows the improved result brought about by bandpass mapping between 100 kHz and 300 kHz. Even with bandpass mapping, there is still severe ringing and broadening in the impulse signal caused by the lack of information in the frequency band with zero energy.

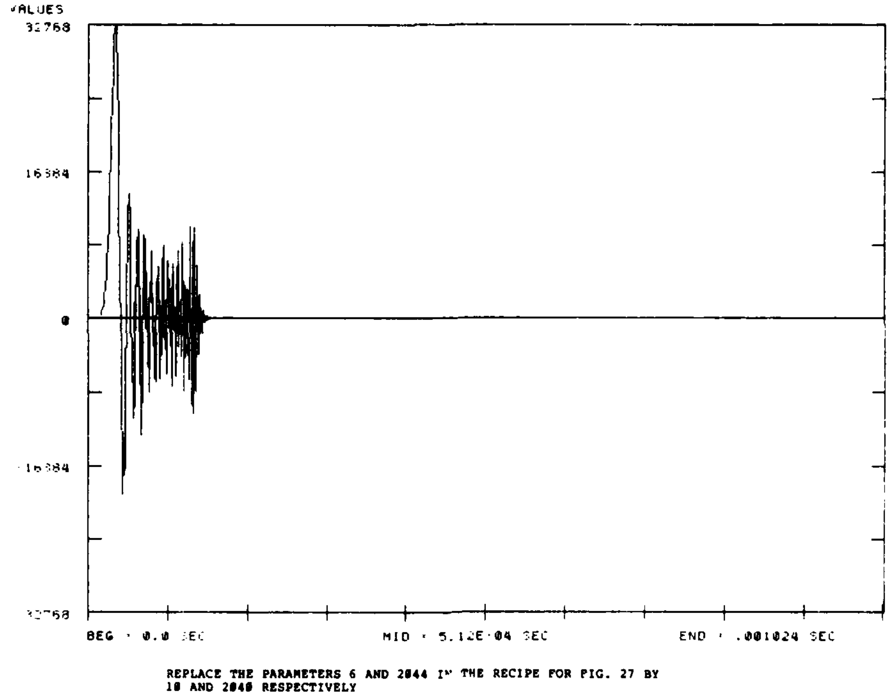


Figure 28. The result of low-time pass filtering the cepstrum of the waveform in Figure 10, $\alpha = 0.999$, width of cepstral filter = $\pm 4.5\ \mu s$

Figure 51 indicates that the band-limited nature of the waveforms recorded using resonance transducers gives rise to severe distortions in the impulses obtained by low-pass filtering the cepstrum of the averaged waveform in Figure 50. Bandpass mapping cleans up the result considerably by suppressing the error propagation in the unoccupied frequency regions, but severe ringing and broadening is still present as shown in Figure 52. This ringing and broadening is due to intrinsic limitations of the band-limited nature of the data—resulting from the use of resonance transducers—and therefore cannot be removed by signal processing.

4.7 Pattern Recognition

The quality of the homomorphically deconvolved signals obtained above does not give one much hope for accomplishing source identification. This observation was confirmed in the following pattern recognition analysis using two sets of waveforms from different sources, with nine waveforms in each set. In one set are waveforms generated by rubbing sandpaper on the edge of one end of the aluminum block, while the ones in the other set were generated by rubbing the sandpaper on the face of the same end of the block. The waveforms were sampled at 2 MHz and are 2048 points in length. Each of the waveforms was homomorphically deconvolved to recover the low-time component in the cepstral domain, using a weighting of 0.9985 and a cepstral cutoff time of $\pm 4.5\ \mu s$. These deconvolved signals were Fourier analyzed, and the 20 Fourier coefficients from 976.56 Hz to 594.7 kHz at 31.25 kHz intervals were extracted. Principal component analysis[6] was performed on these 20 Fourier coefficients of

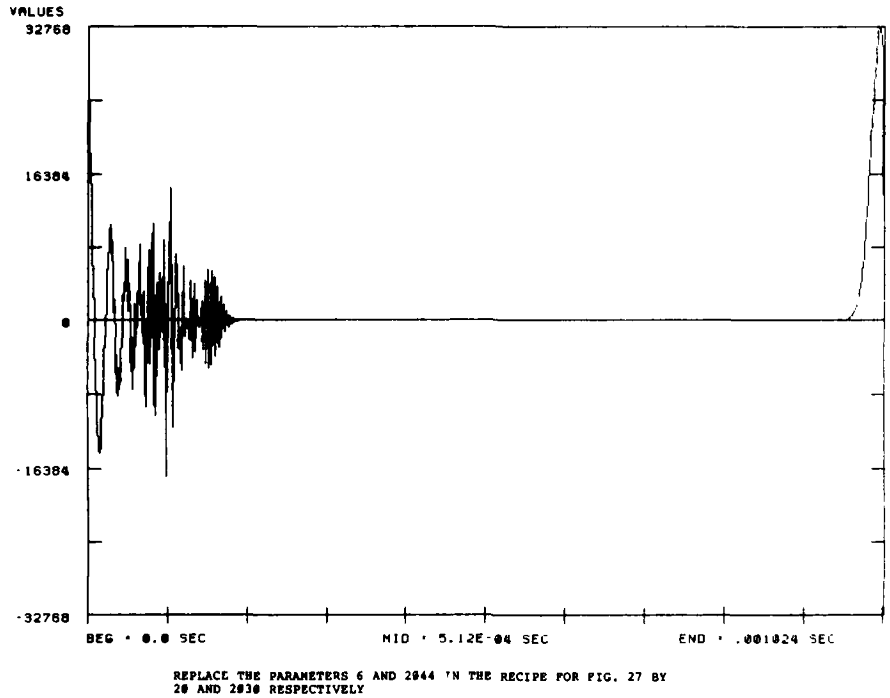
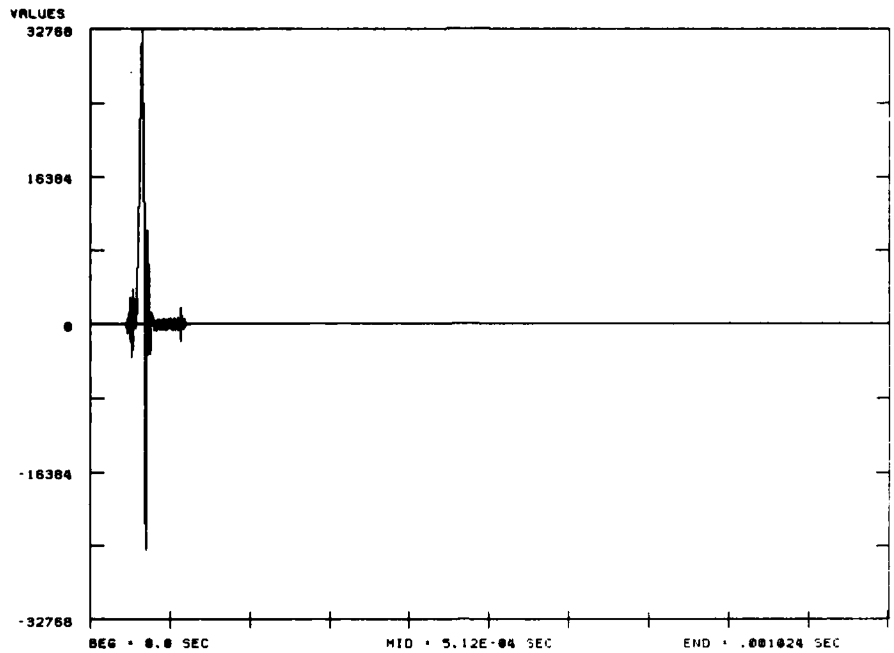


Figure 29. The result of low-time pass filtering the cepstrum of the waveform in Figure 10, $\alpha = 0.999$, width of cepstral filter = $\pm 9.5\ \mu s$

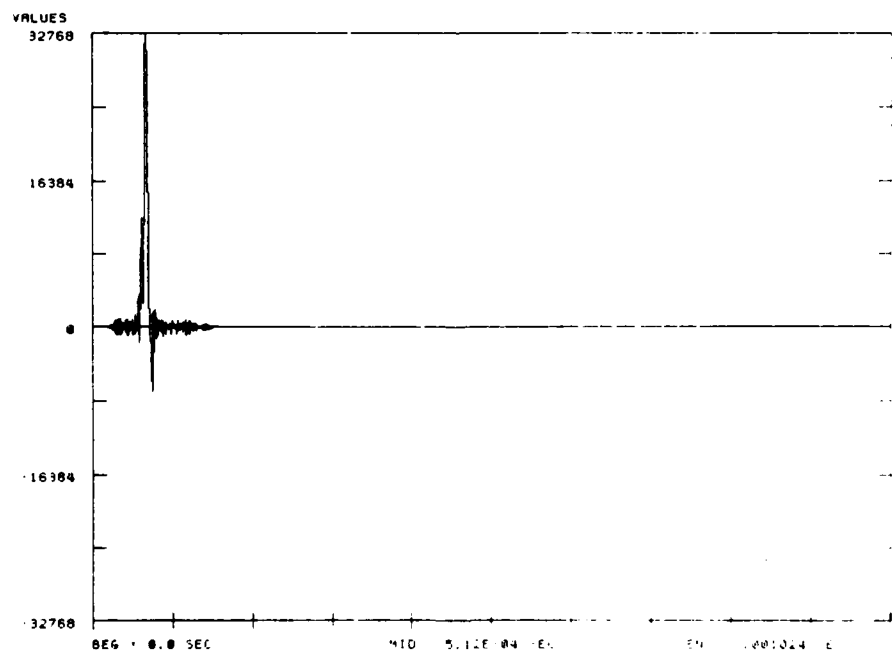
the two sets of 9 waveforms in each set using the software in the ILS signal analysis package. The first two principal components are plotted in Figure 53. (See Appendix C for a description on pattern recognition and examples of command procedures for Figures 53 and following.) The results in Figure 53 indicate the two sets of waveforms cannot be distinguished from each other. In comparison, the first two principal components of the raw data without cepstral filtering are plotted in Figure 54. The two sets of waveforms become less separated after the cepstral filtering.

Similar results were obtained when the above procedure was repeated with an additional set of nine lead-breaking waveforms. The plots of the first two principal components with and without cepstral filtering are illustrated in Figures 55 and 56. Again, cepstral filtering failed to improve the separability of the three sets of waveforms.



REPLACE THE PARAMETERS 10 AND 8184 IN THE RECIPE FOR FIG. 19 BY
6 AND 8188 RESPECTIVELY

Figure 30. The result of low-time pass filtering the cepstrum of the waveform in Figure 15, $\alpha = 0.9985$, width of cepstral filter = $\pm 2.5\ \mu s$



SAME AS FIG. 19

Figure 31. The result of low-time pass filtering the cepstrum of the waveform in Figure 15, $\alpha = 0.9985$, width of cepstral filter = $\pm 4.5\ \mu s$

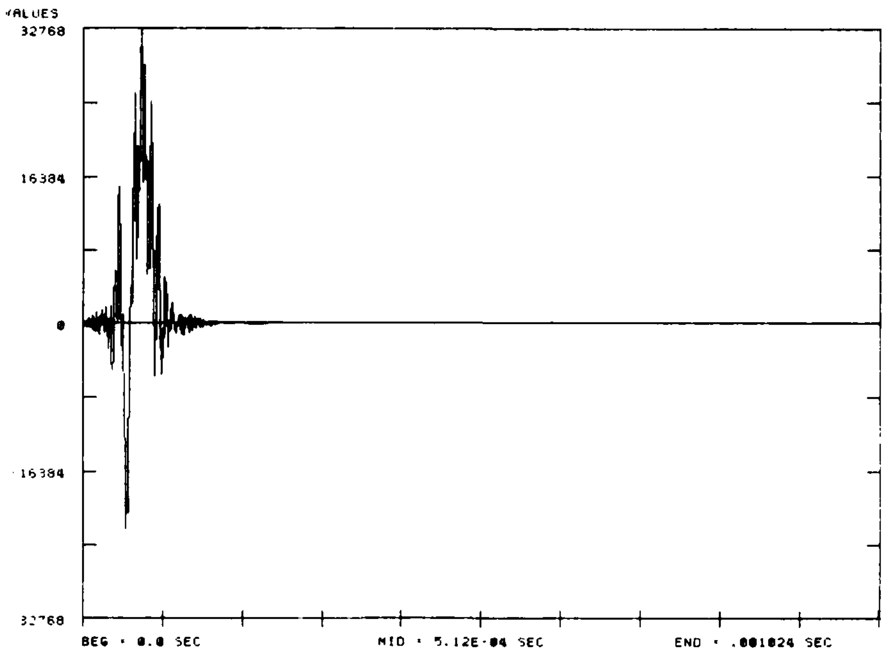


Figure 32. The result of low-time pass filtering the cepstrum of the waveform in Figure 15, $\alpha = 0.9985$, width of cepstral filter = $\pm 9.5\ \mu s$

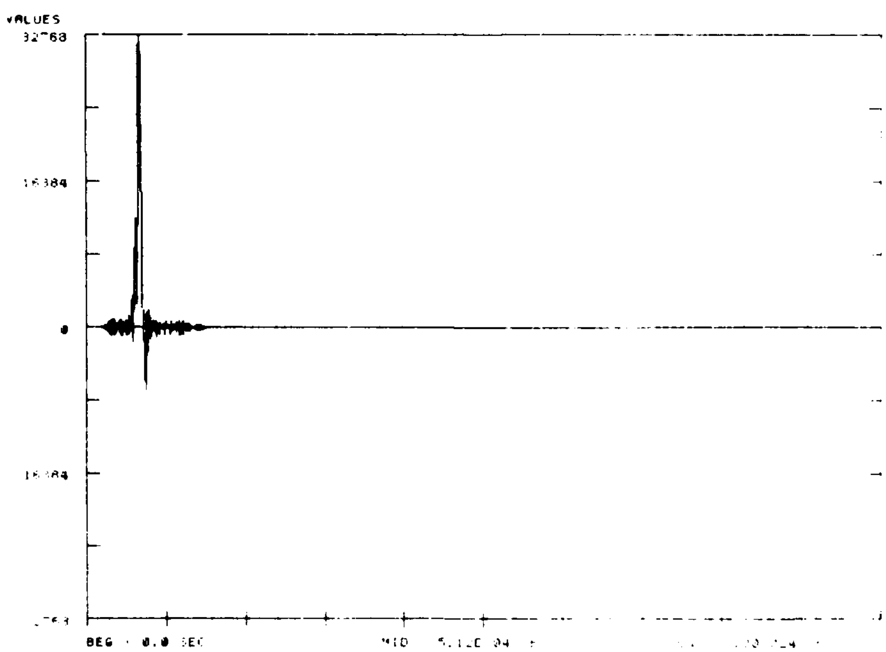
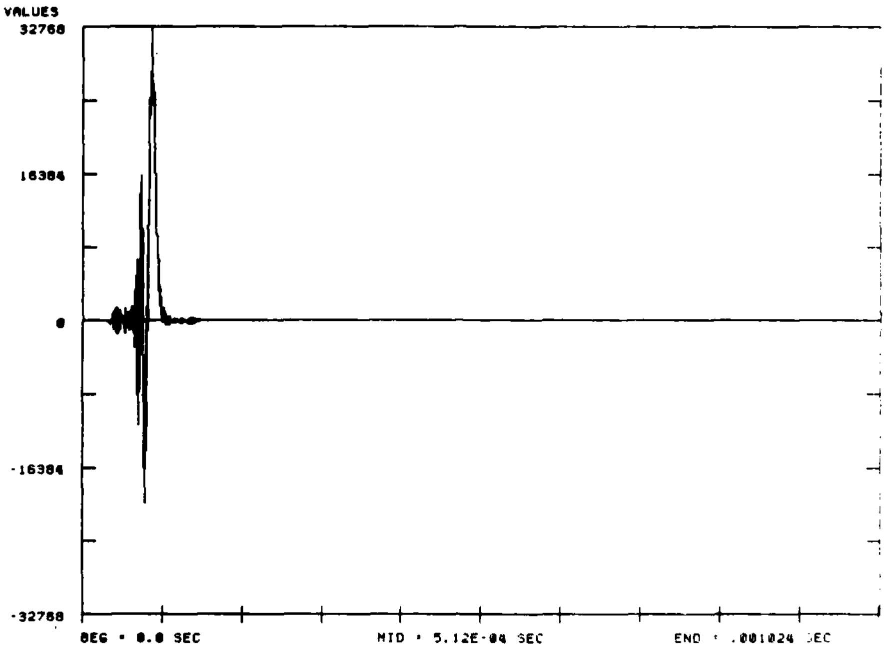
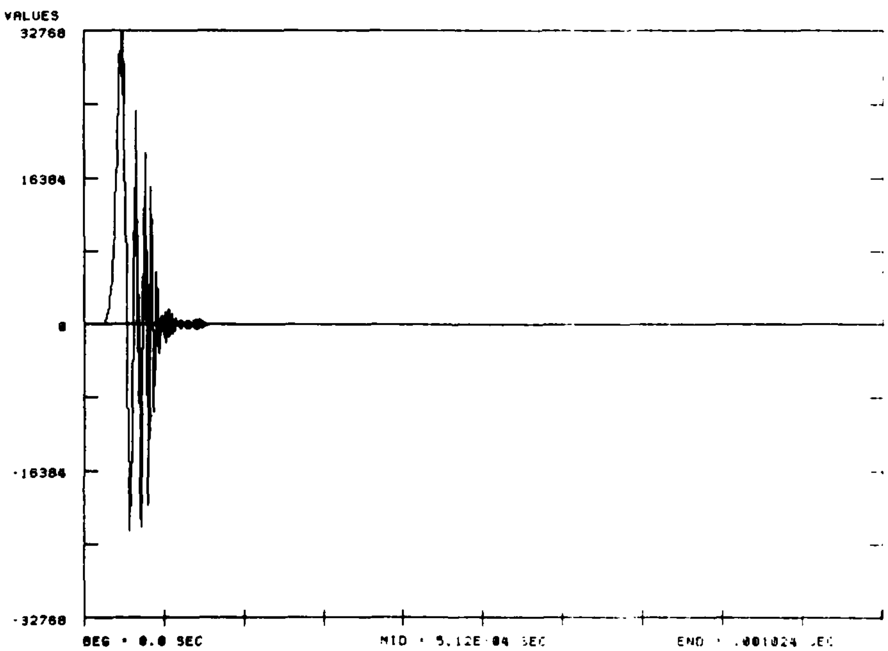


Figure 33. The result of low-time pass filtering the cepstrum of the waveform in Figure 15, $\alpha = 0.9985$



REPLACE THE PARAMETER 0.9985 IN THE RECIPE FOR FIG. 19 BY 0.998

Figure 34. The result of low-time pass filtering the cepstrum of the waveform in Figure 15, $\alpha = 0.9980$



REPLACE THE PARAMETER 0.9985 IN THE RECIPE FOR FIG. 19 BY 0.9975

Figure 35. The result of low-time pass filtering the cepstrum of the waveform in Figure 15, $\alpha = 0.9975$

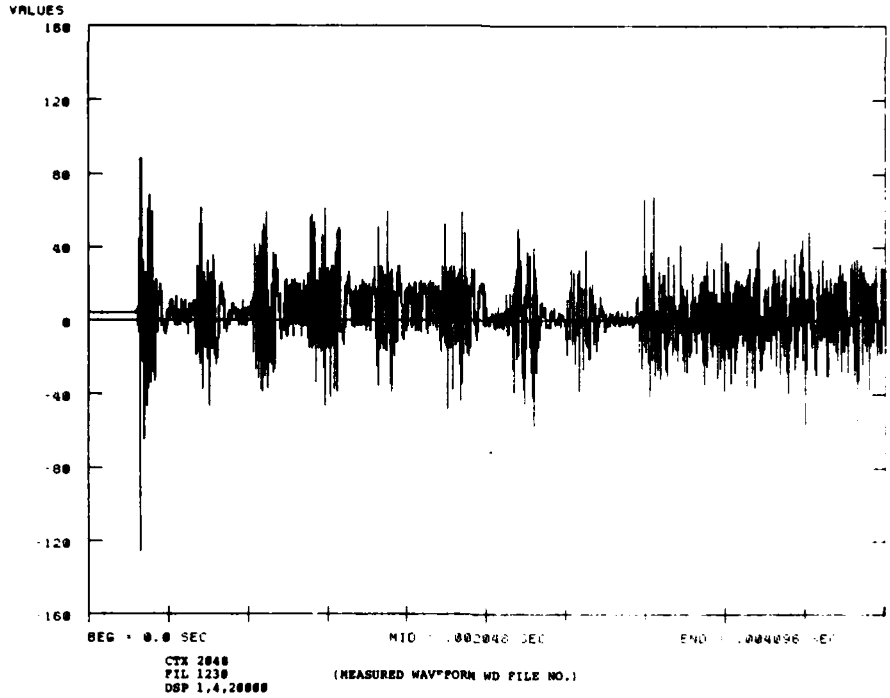


Figure 36. A typical large-angle Pentel lead-break waveform; the angle between the lead and the surface is 65°

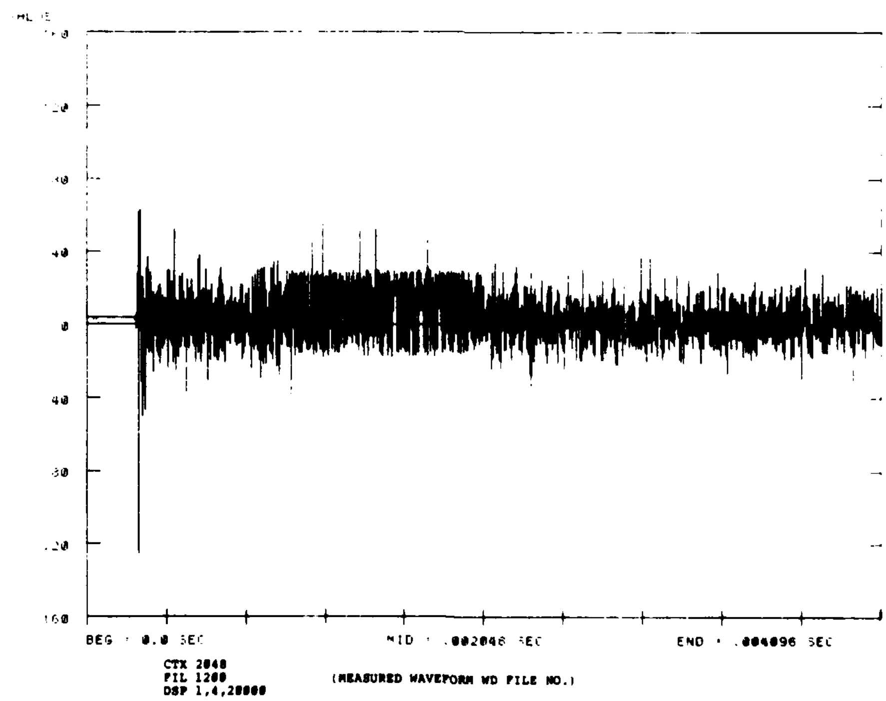


Figure 37. A typical small-angle Pentel lead-break waveform; the angle between the lead and the surface is 40°

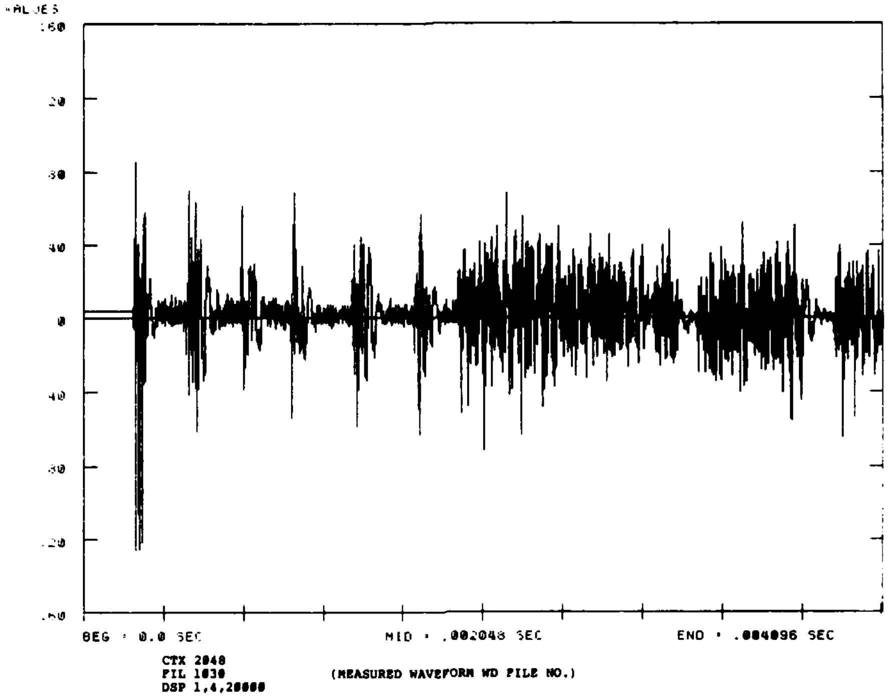


Figure 38. A Pentel lead-break waveform with secondary Pentel tip impact at a large angle

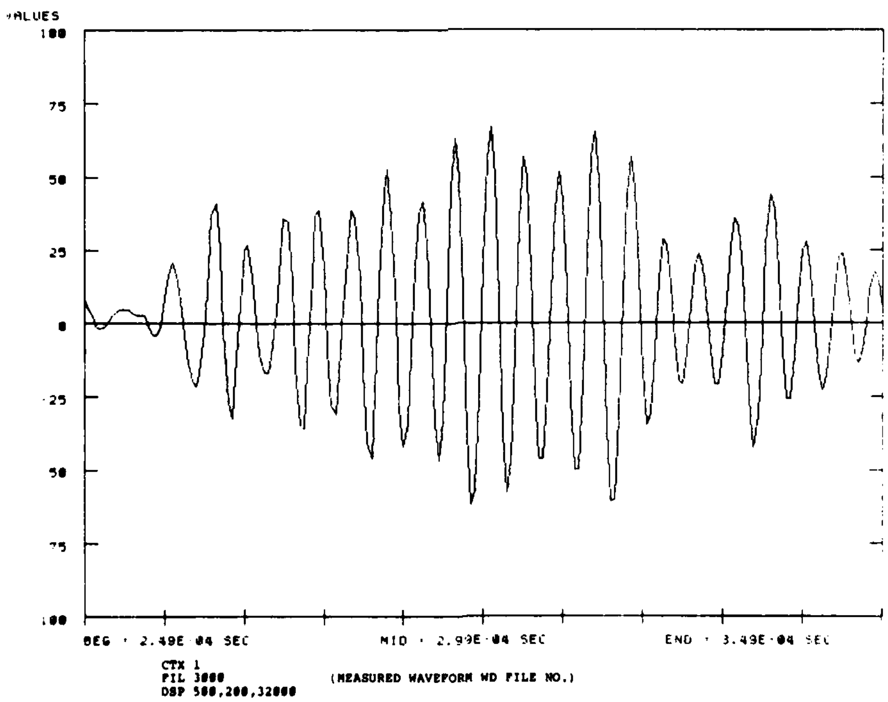


Figure 39. A 200-point sample of a fixture-generated lead-break waveform near the initial impulse

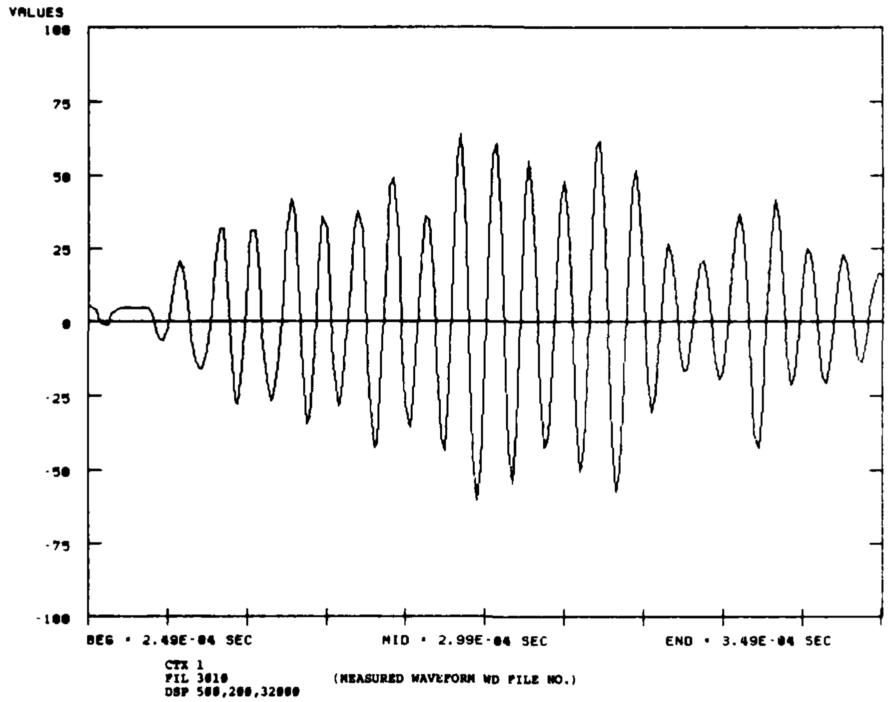


Figure 40. A 200-point sample of another fixture-generated lead-break waveform near the initial impulse

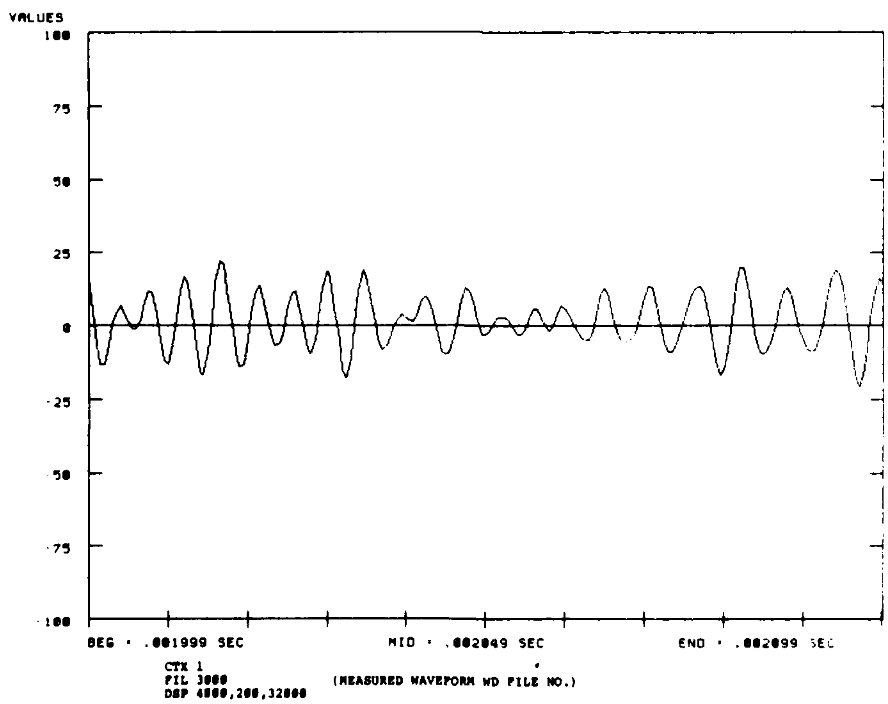


Figure 41. Another 200-point sample of the waveform in Figure 39 near the later part of the waveform

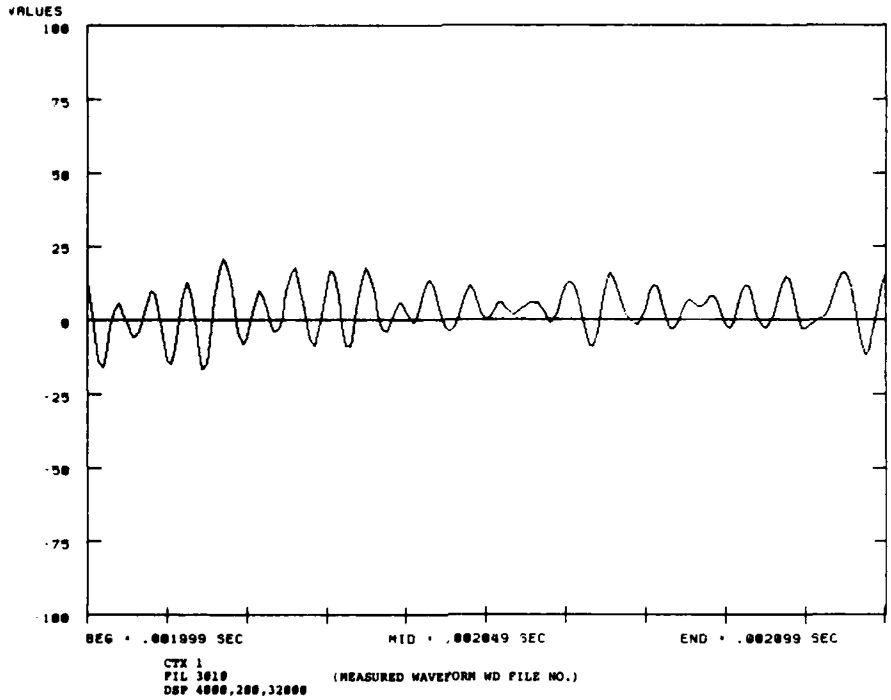
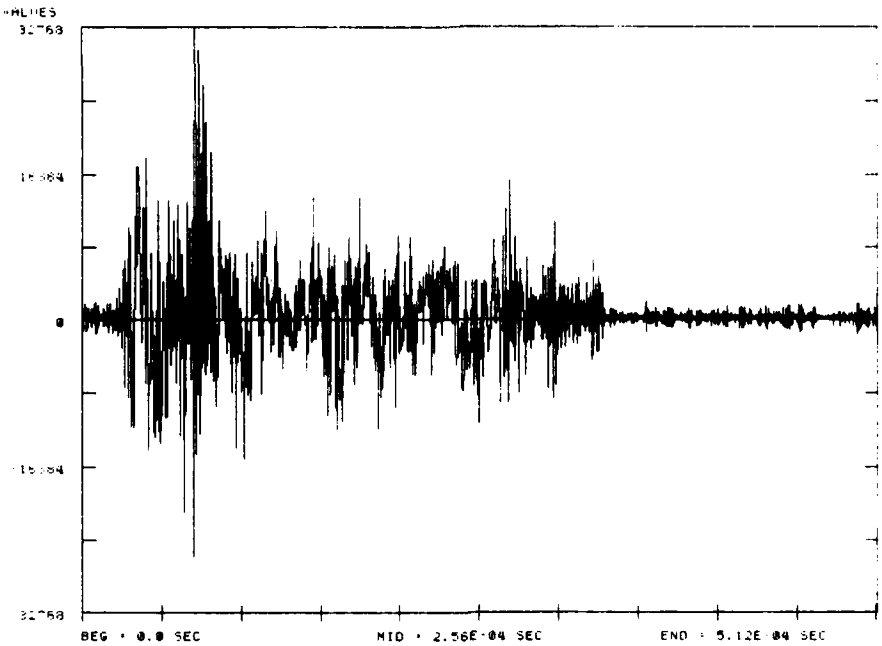
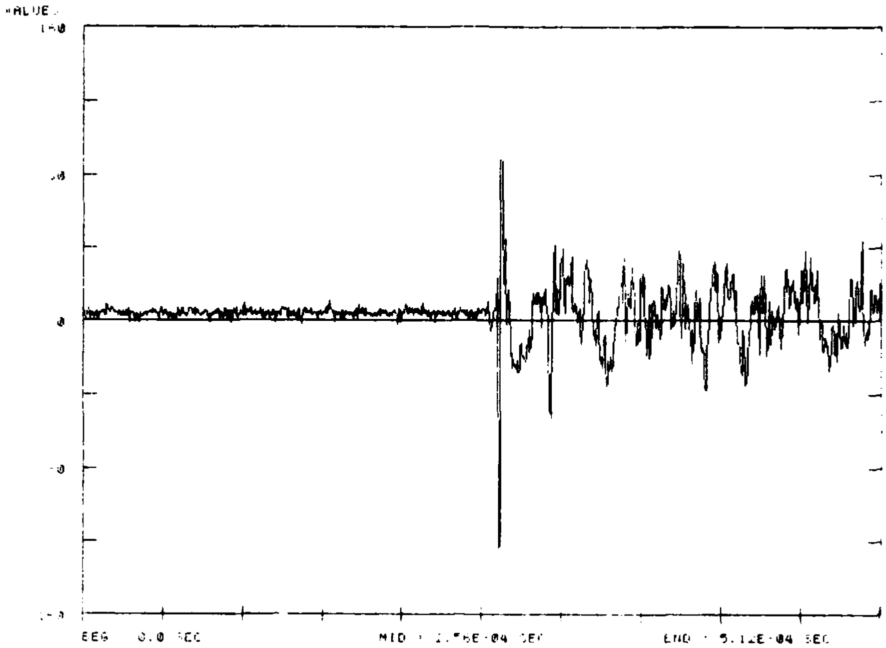


Figure 42. Another 200-point sample of the waveform in Figure 40 near the later part of the waveform



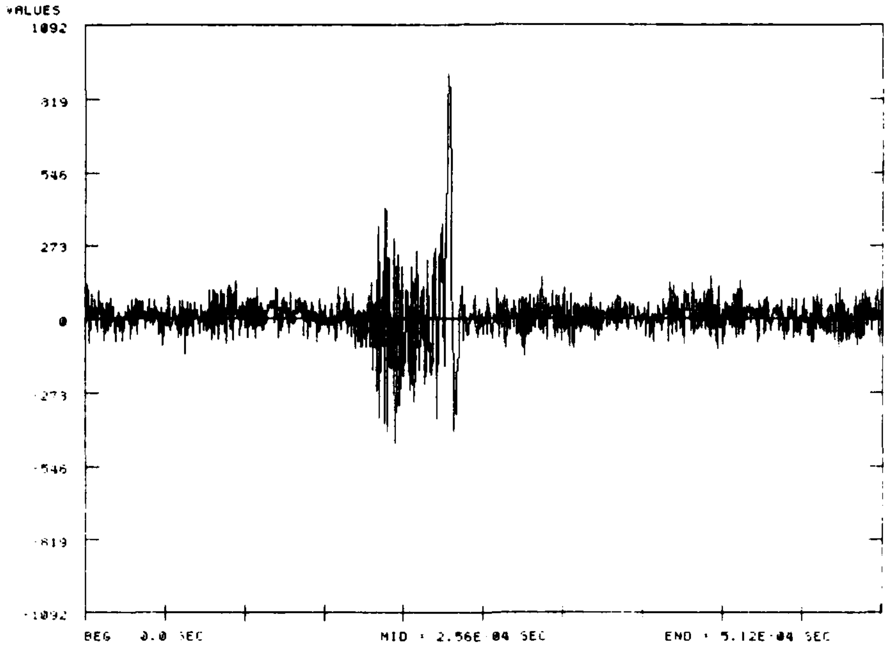

```
CTX 6192
@AVERG                  (AVERAGE A NUMBER OF WAVEFORMS)
>FN  (INPUT FILE NAME CONTAINING THE WD NOS. OF THE FILES TO
      BE AVERAGED)
>N1  (OUTPUT SECONDARY WD FILE NO. TO STORE THE AVERAGED WAVEFORM)
CTX 1024
FIL N1
FIL SO
>CZN2,,1,1024           (N2 IS THE SECONDARY WD FILE NO., 0<N2<9990,
                         TO STORE THE FIRST 1024 CHANNELS OF FILE N1)

MDF S0,63,-32000
TRF H
TRF 1,1,1,100           (COPY THE FIRST 1024 CHANNELS OF FILE N1 TO
                         FILE N2)
FIL N2
FIL SN3                 (N3 IS THE SECONDARY WD FILE NO., 0<N3<9990,
                         TO STORE THE CEPSTRUM)
XCP,,1
>0.999
@XCPIHIGH               (HIGH-PASS CEPSTRAL FILTER AND
                         INVERSE TRANSFORM)
>N3
>N4                     (N4 IS THE SECONDARY WD FILE NO., 0<N4<9990,
                         TO STORE THE INVERSE CEPSTRUM)
>9
>1017
>1024
DSP S1,1,100
```

Figure 43. The impulse response obtained by high-time pass filtering the cepstrum of the average of 33 mechanically generated lead-break waveforms; α = 0.999



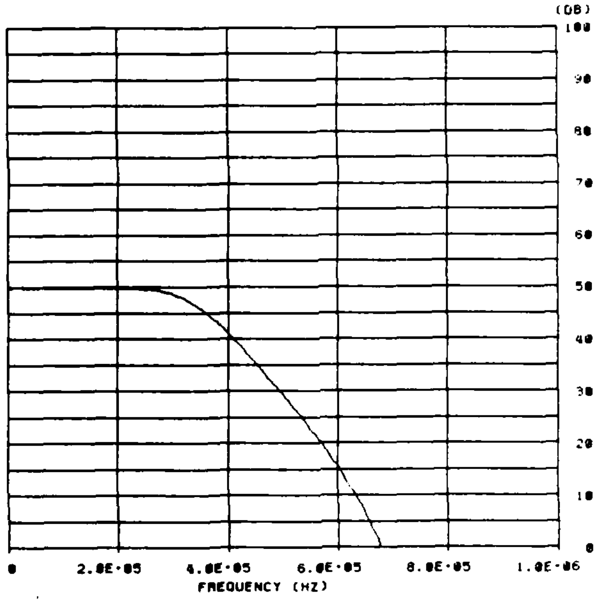

```
CTX 1024
FIL 3200              (MEASURED WAVEFORM WD FILE NO.)
FIL SO
>CZN1,,1,1024         (N1 IS THE SECONDARY WD FILE NO., 0<N1<9990,
                       TO STORE THE FIRST 1024 CHANNELS OF FILE 3200)

MDF S0,63,-32000
TRF H
TRF 1,1,1,100        (COPY THE FIRST 1024 CHANNELS OF FILE 3200 TO
                      FILE N1)
DSP S1,1,20000
```

Figure 44. One of the 33 mechanically generated lead-break waveforms used in the averaging



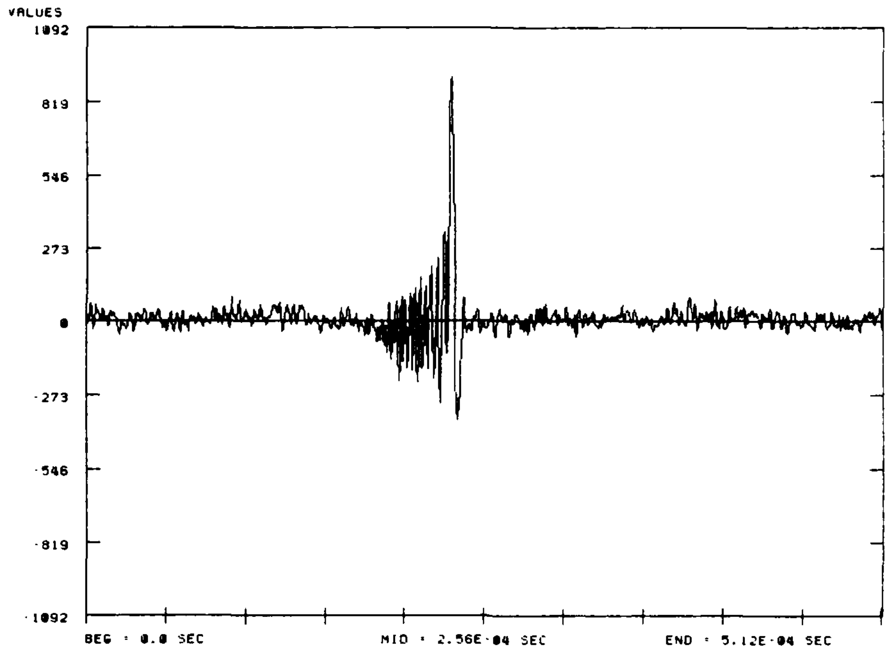

```
CTX 1024
@FDECON        (FOURIER DECONVOLUTION)
>N1            (WD NO. OF FIG. 44, WAVEFORM TO BE DECONVOLVED)
>N2            (WD NO. OF FIG. 43, DECONVOLVING KERNEL)
>N3            (WD NO. FOR THE SECONDARY FILE TO STORE THE
                DECONVOLVED RESULT)
DSP S1,1,3000
```

Figure 45. The result of deconvolving the waveform in Figure 44 by the impulse response in Figure 43




```
EFI 4,0,2000000,0,700000,-60     (DESIGNING AN ELLIPTICAL FILTER
                                  WITH 4 POLES, FLAT PASS BAND,
                                  2 MHZ SAMPLING FREQUENCY,
                                  PASS BAND FROM 0 TO 700 KHZ,,
                                  STOPBAND ATTENUATION 60DB0
TPU G4                            (PLOT OUT THE FILTER)
```

Figure 46. An elliptical frequency filter used to reduce the high-frequency noise in the result in Figure 45; −60 dB cut-off at 700 kHz




```
CTX 1024
FIL N1          (WD NO. OF FIG. 45)
FIL SN2         (N2 IS THE WD NO. FOR THE SECONDARY FILE TO
                 STORE THE FILTERED RESULT)
EFI 4,0,2000000,0,700000,-60    (SEE THE DESCRIPTION IN FIG. 46)
FLT E1,1                        (FILTER FILE N1 WITH THE DESIGNED
                                 FILTER)
DSP S1,1,100
```

Figure 47. The result of filtering the waveform in Figure 45 with the elliptical filter in Figure 46

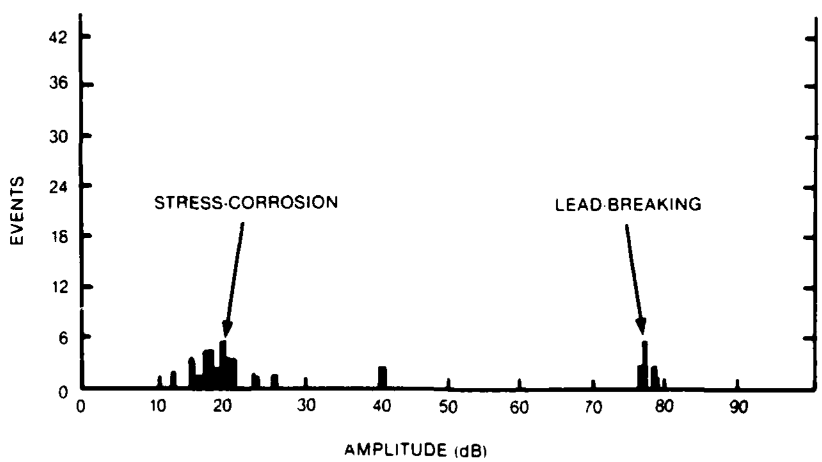
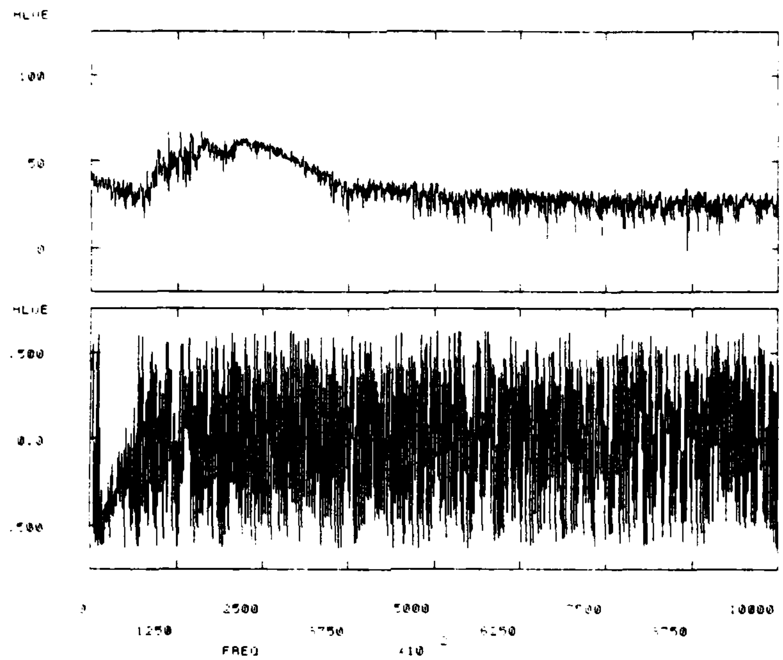
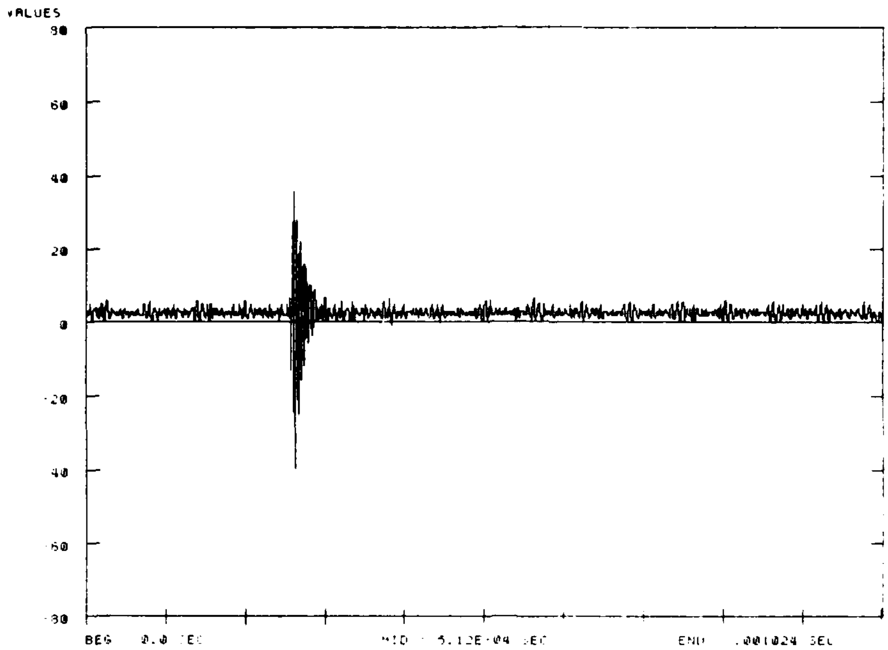


Figure 48. Relative amplitude of the stress corrosion events and the lead-breaking events



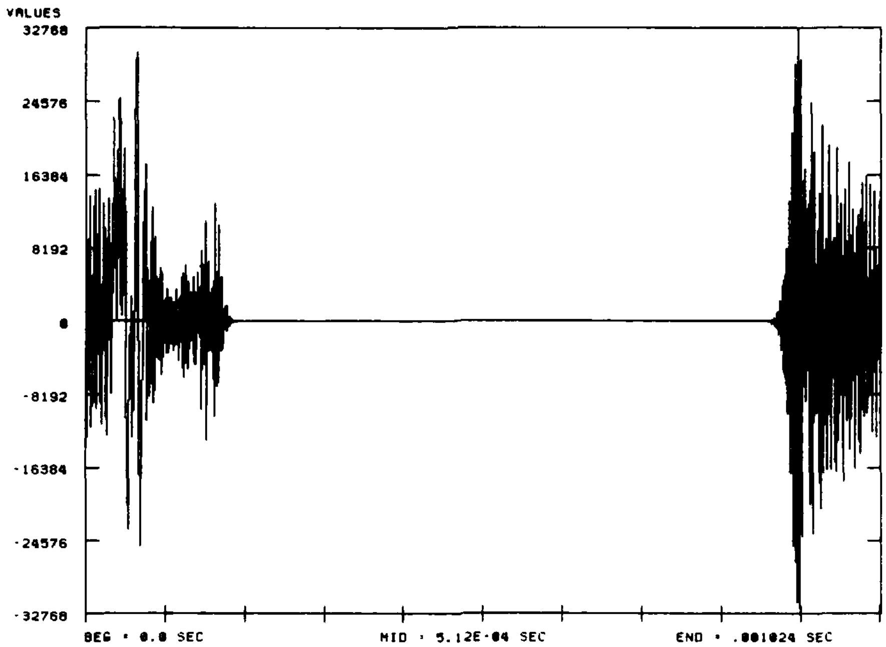

```
CTX 2048
FIL 6500        (WD NO. OF THE MEASURED WAVEFORM)
FIL SO
>CZN1,,1,2048   (N1 IS THE WD NO. FOR THE SECONDARY FILE TO STORE
                 THE FIRST 2048 CHANNELS OF THE MEASURED WAVEFORM)

MDF S0,63,-32000
TRF H
TRF 1,1,1,100   (COPY THE FIRST 2048 CHANNELS OF FILE 6500 TO
                 FILE N1)
FIL N1
FIL SN2         (N2 IS THE WD NO. FOR THE SECONDARY FILE TO STORE
                 THE RECORD VERSION OF N1)
OPN S1
SRE 1,1         (CONVERT FILE N1 FROM INTEGER TO REAL NUMBER
                 AND STORE THE RESULT IN FILE N2)
FIL N2
FIL SN3         (N3 IS THE WD NO. FOR THE SECONDARY FILE TO STORE
                 THE FOURIER TRANSFORM OF N2)
FFT P           (CALCULATE THE FOURIER TRANSFORM IN POLAR COORDINATE)
DRE S           (DISPLAY THE FOURIER TRANSFORM)
```

Figure 49. Power spectrum of a stress-corrosion event captured by a resonance transducer



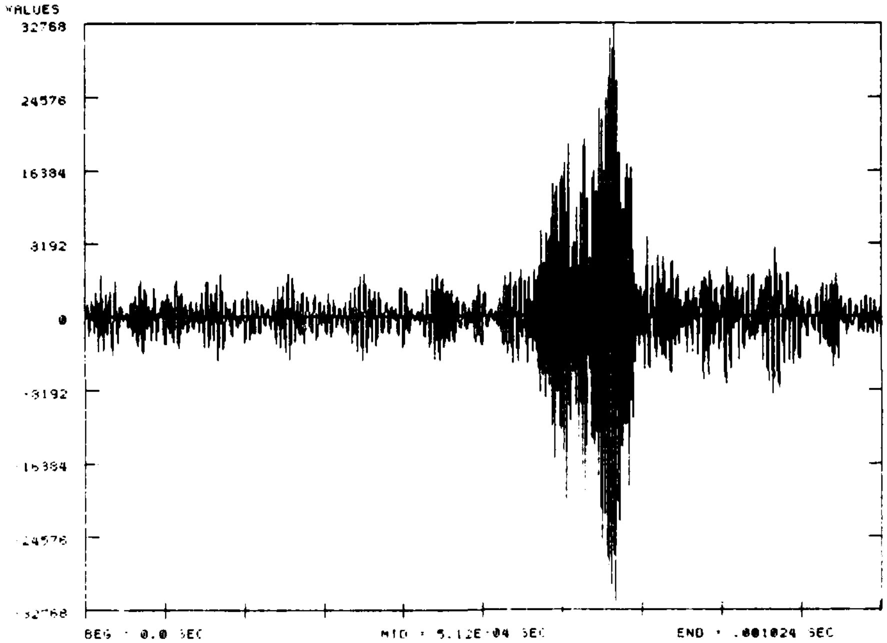

```
CTX 8192
@AVERG          (SEE THE DESCRIPTION IN FIG. 43)
>FN  (INPUT FILE NAME CONTAINING THE WD NOS. OF THE FILES TO
      BE AVERAGED)
>N1  (OUTPUT SECONDARY WD FILE NO. TO STORE THE AVERAGED WAVEFORM)
CTX 2048
FIL N1
FIL SO
>CZN2,,1,2048       (N2 IS THE SECONDARY WD FILE NO., 0<N2<9990,
                     TO STORE THE FIRST 2048 CHANNELS OF FILE N1)

MDF S0,63,-32000
TRF H
TRF 1,1,1,100       (COPY THE FIRST 2048 CHANNELS OF FILE N1 TO
                     FILE N2)
DSP S1,1,40000
```

Figure 50. The average of 31 stress-corrosion waveforms with their initial impulses lined up at the same location; the waveforms were captured with a resonance transducer




```
CTX 2048
FIL N1              (WD NO. OF FIG. 50)
FIL SN2             (N2 IS THE SECONDARY WD FILE NO., 0<N2<9990,
                     TO STORE THE CEPSTRUM OF N1)
XCP,,1              (SEE THE DESCRIPTION IN FIG. 13)
>0.999
@XCPILOW            (SEE THE DESCRIPTION IN FIG. 12)
>N2
>N3                (N3 IS THE SECONDARY WD FILE NO., 0<N2<9990,
                    TO STORE THE INVERSE CEPSTRUM)
>10
>2040
>2048
DSP S1,1,100
```

Figure 51. The result of low-time pass filtering the cepstrum of the averaged waveform in Figure 50; no band-pass mapping




```
CTX 2048
FIL N1                (WD NO. OF FIG. 50)
FIL SN2               (N2 IS THE SECONDARY WD FILE NO., 0<N2<9990,
                       TO STORE THE CEPSTRUM OF N1)
PCP P,,1,100000,300000  (TAKE THE CEPSTRUM WITH EXPONENTIAL
                         WEIGHTING AND BAND PASS MAPPING
                         FROM 100 KHZ TO 300 KHZ)
>0.999               (SEE THE DESCRIPTION IN FIG. 13)
@XCPIPLOW            (SEE THE DESCRIPTION IN FIG. 12)
>N2
>N3                  (N3 IS THE SECONDARY WD FILE NO., 0<N2<9990,
                      TO STORE THE INVERSE CEPSTRUM)
>10
>2040
>2048
DSP S1,1,100
```

Figure 52. The result of low-time pass filtering the cepstrum of the averaged waveform in Figure 50; band-pass mapping between 100 kHz and 300 kHz

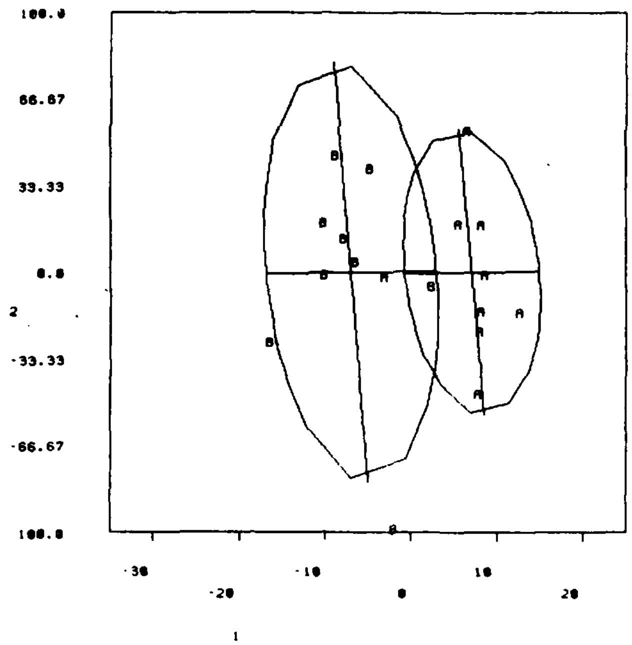


Figure 53. **Scatter plots of the first two principal components of the frequency components of two sets of events after low-time pass filtering in the cepstral domain. The two sets of events were generated by (a) rubbing a sandpaper on the edge of one end of the aluminum block and (b) rubbing the sandpaper on the facet of the same end of the block**

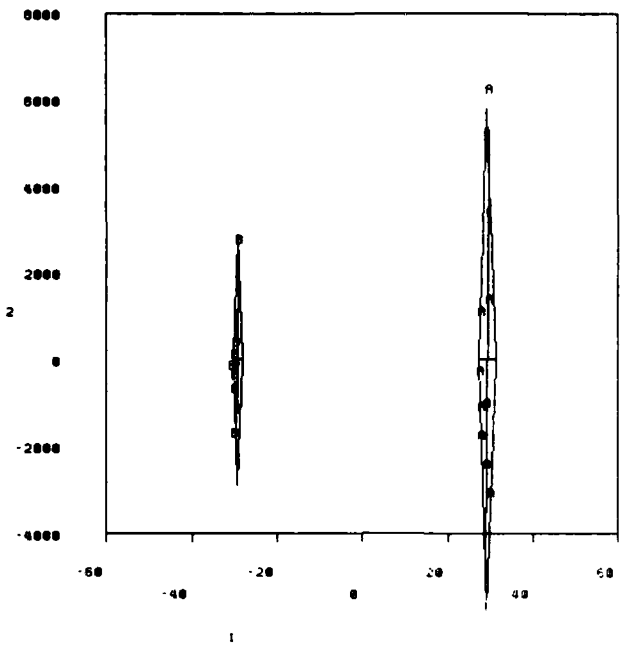


Figure 54. Scatter plots of the first two principal components of the frequency components of the two sets of events in Figure 53 without filtering in the cepstral domain

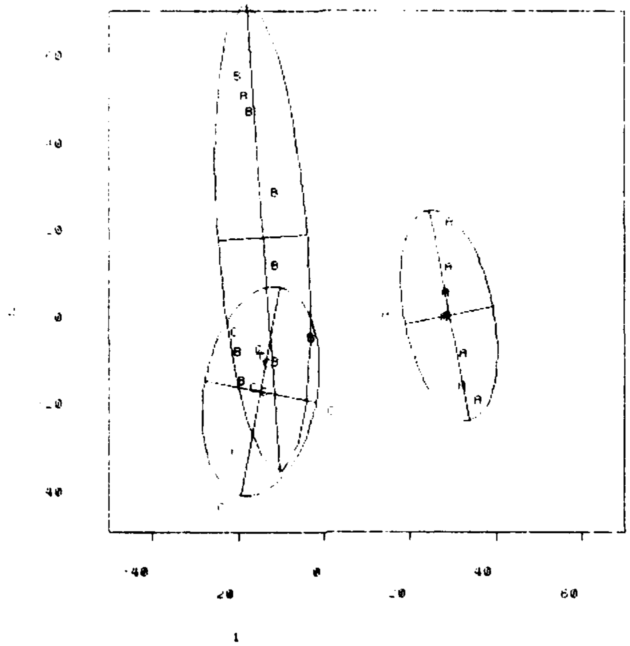


Figure 55. Scatter plots of the first two principal components of the frequency components of three sets of events after low-time pass filtering in the cepstral domain; the three sets of events were generated by (a) rubbing a sandpaper on the edge of one end of the aluminum block, (b) rubbing the sandpaper on the facet of the same end of the block, and (c) lead-breaking

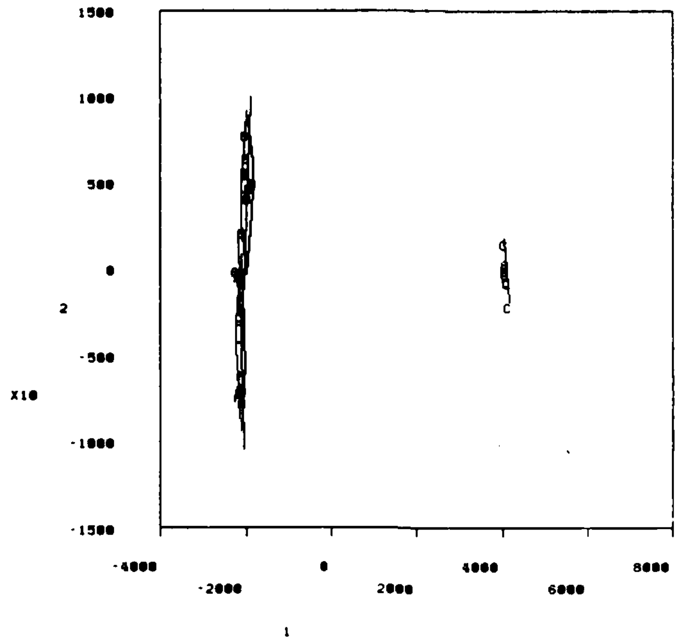


Figure 56. Scatter plots of the first two principal components of the frequency components of the three sets of events in Figure 55 without filtering in the cepstral domain

Section 5

CONCLUSION

The key results obtained in this study are summarized below:

- Adaptive homomorphic deconvolution seems to work relatively well for AE signals captured with a wide-band flat response transducer with good S/N. This technique is able to clean up multipath and multimode reverberation and ringdown effects rather well. The technique does, however, have some unpleasant ad hoc elements, in that several key parameters cannot be determined *a priori.* Specifically, the decay exponent, needed for exponential weighting to make the sequence a well-behaved minimum phase sequence, is a problem, as are the linear filter coefficients needed for filtering the cepstral waveform. The appropriate constants can quickly be found with a little experimentation, but the process is non-unique.
- Adaptive homomorphic deconvolution seems to work to some degree with a conventional resonant transducer, particularly when bandpass mapping is employed. The success of the deconvolution is severely limited, however, because of the limited signal bandwidth information provided by the resonant transducer. As can be seen in plots shown in Section 4.6, although pulse compression has been achieved, the reconstructed signal suffers from serious ringing effects due to the very limited transducer bandwidth.
- Adaptive homomorphic deconvolution, implemented by high-pass filtering the averaged cepstral domain signal, seems to be effective in providing approximate estimates of the workpiece transfer function, particularly when the original signal is obtained with a wide-band transducer. As noted above, the technique is not effective when narrow-band resonant transducers are employed, because of the limited illumination of the cepstral domain afforded by those transducers. Unfortunately, the wide-band transducers employed in this study are not suitable for use in practical applications because of their fragility and severely limited sensitivity. Their sensitivity is not high enough to be used to obtain signals from typical, real AE sources; specifically, they are too insensitive for use with aluminum stress corrosion V-block sources, or most practical AE sources in general.
- The final step in employing adaptive homomorphic deconvolution to analyze acoustic emission signals is to employ ordinary Fourier deconvolution to eliminate the effects of transducer ringdown, multipath, etc. Fourier deconvolution, when used to eliminate the workpiece transfer function effects, is very sensitive to the precise details of the transfer function. In practical terms, this means that the transfer functions must be precisely repeatable from pulse to pulse. In fact, the data obtained in this study leads one to believe that there are substantial variations in the transfer function obtained from both artificial and natural AE sources, particularly when relatively long records of the pulse ringdown signals are obtained. The transfer functions are more repeatable when relatively short records are used. Preliminary analysis suggests that slight variations in the direction of the signal propagation may be responsible for these effects. A slight variation in the direction of propagation, or in the mix of longitudinal and shear

waves, will result in changes in the workpiece ringdown response function; these changes get larger and larger as the ringdown proceeds. Thus, the later segments of long records will suffer progressively increasing distortion from inaccurate deconvolution. Natural AE signals, since they emanate from evolving cracks where the crack propagation follows natural grain boundaries, may be expected to offer some variation in the strength and direction of the emitted shear and longitudinal waves. Surprisingly, even AE signals carefully generated via artificial means (Pental lead breaks) seem to vary in the precise details of their ringdown responses over relatively long record lengths. The net result is that only relatively short (~2048 points) record lengths can be used successfully, either with natural or artificial AE signal sources.

- Pattern recognition, employed as a means of identifying and characterizing AE sources through differences in the microstructure of their waveforms, does not seem to be particularly effective, either when applied to the original waveforms, or when applied to waveforms restored using adaptive homomorphic deconvolution. The original waveforms are heavily contaminated with the workpiece transfer function effects, and microstructure effects are difficult to separate. The reconstructed waveforms suffered from ringing distortion because of the limited bandwidth of the transducers employed. They were also very difficult to separate.

In conclusion, the analytical techniques explored in this research provide several new options in processing acoustic emission signals. Specifically, they provide the means to obtain and employ workpiece transfer functions in deriving the underlying acoustic emission waveforms. Successful application of these techniques requires the use of wide-band transducers that are faithful to the original waveforms and can only be employed on relatively short records. Any practical application must await the development of wide-band accurate transducers with much improved sensitivity over currently available transducers.

Section 6

REFERENCES

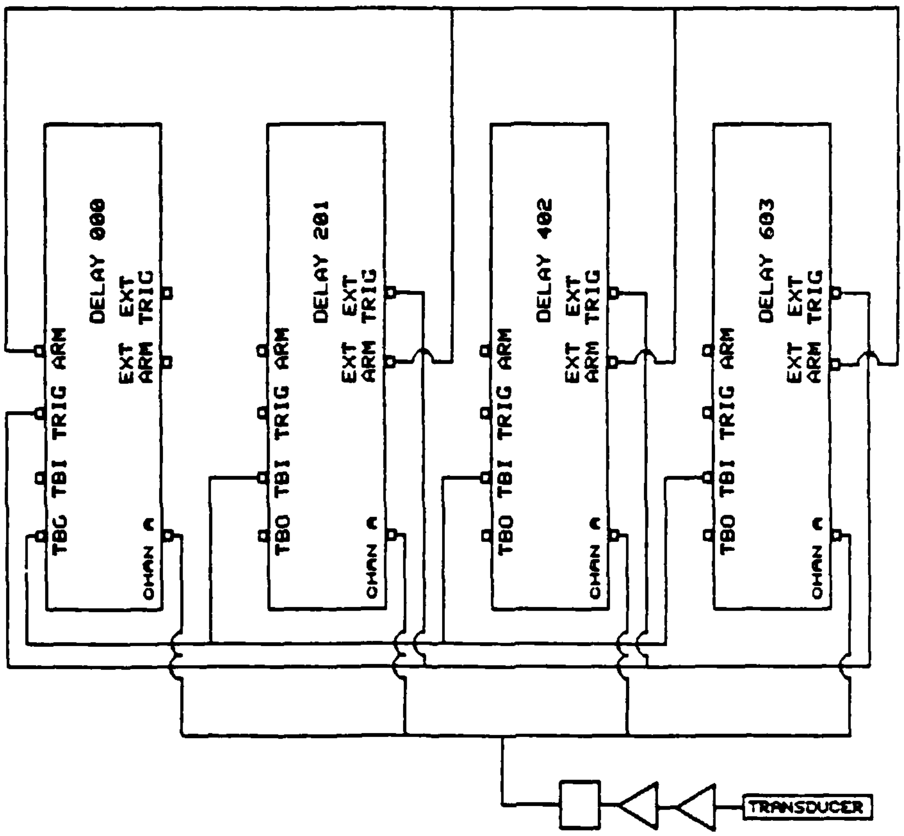
1. Eitzen, D., Breckenridge, F., Clongh, R., Hsa, N., Proctor, T., and Simmons, J., "NBS Developments in Qualatative Acoustic Emission Measurements," *AF/DARPA Review of Qualitative NDE,* August 1981, Boulder, Colorado.
2. Tribolet, J.M., *Seismic Applications of Homomorphic Signal Processing* (Englewood Cliffs, N.J.: Prentice-Hall, 1979).
3. Oppenheim, A.V., and Schaefer, R.W., *Digital Signal Processing* (Englewood Cliffs, N.J.: Prentice-Hall, 1975).
4. Brigham, E.O., *The Fast Fourier Transform* (Englewood Cliffs, N.J.: Prentice-Hall, 1974).
5. Weinstein, Clifford J., et al., *Programs for Digital Signal Processing* (IEEE Press, 0-87942-127-4).
6. Chen, Y.T., *Interactive Pattern Recognition* (New York: Marcel Dekker, 1978).

Appendix A

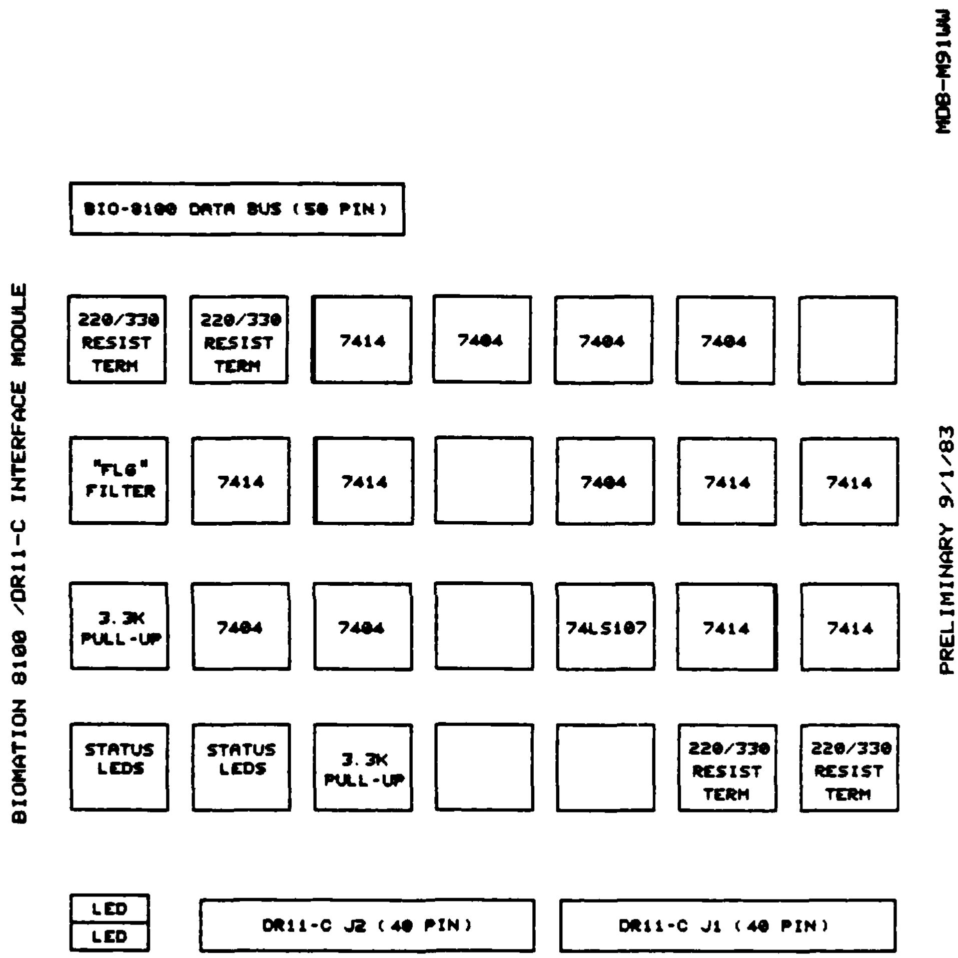
BIOMATION 8100 DR11-C INTERFACE

PRELIMINARY DOCUMENTATION

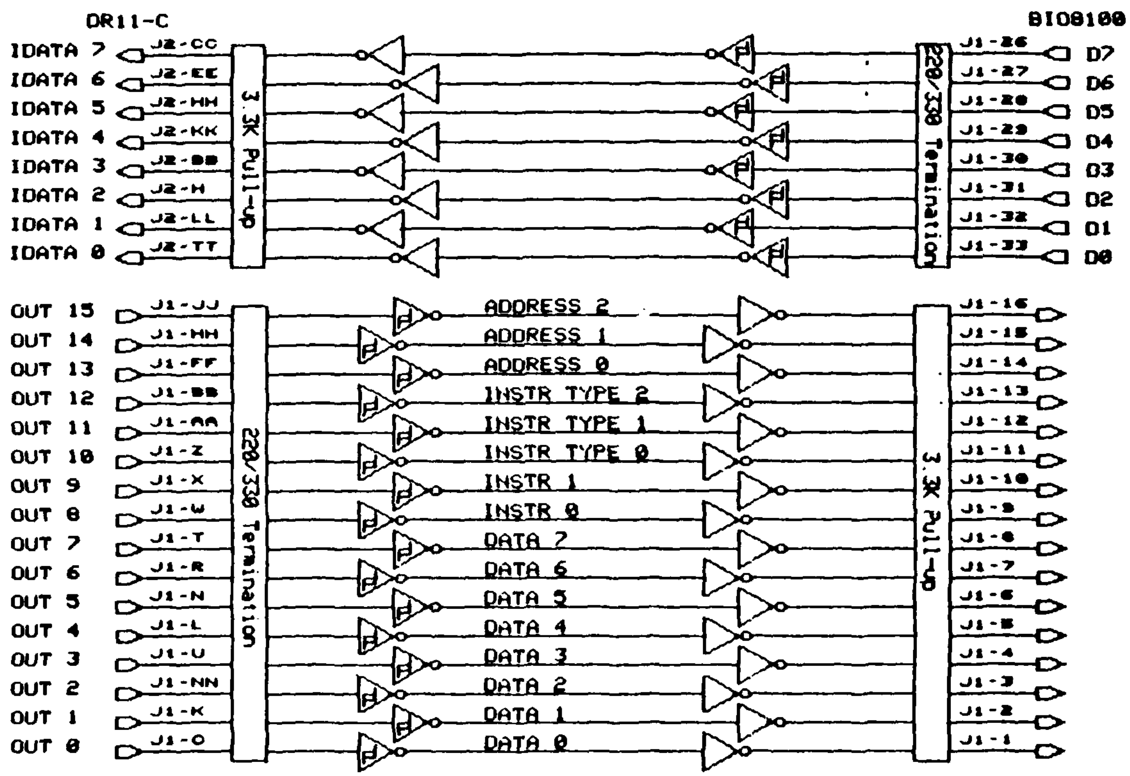
BIOMATION 8100
DR11-C
INTERFACE



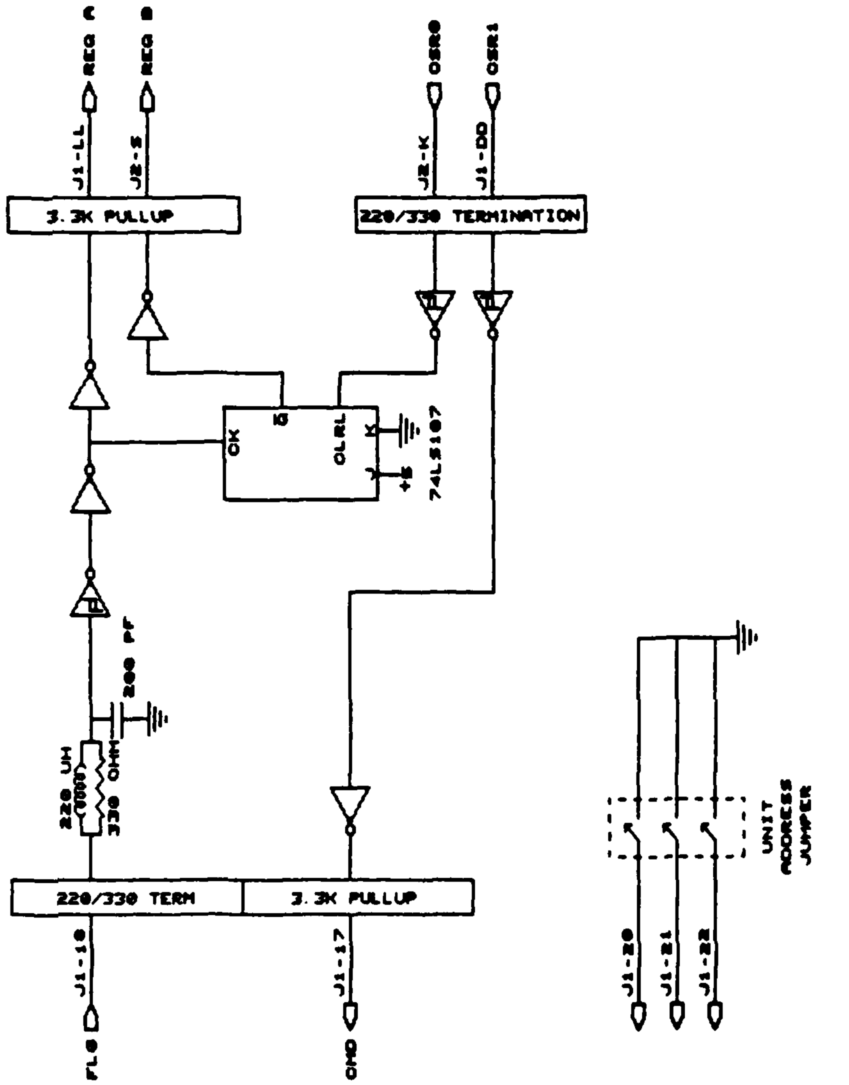
PRELIMINARY 9/1/83


BIOMATION 8100 /DR11-C INTERFACE MODULE
PRELIMINARY 9/1/83
MDB-M91WW
BIO-8100 DATA BUS (50 PIN)
220/330 RESIST TERM
220/330 RESIST TERM
7414
7404
7404
7404
"FLG" FILTER
7414
7414
7404
7414
7414
3.3K PULL-UP
7404
7404
74LS107
7414
7414
STATUS LEDS
STATUS LEDS
3.3K PULL-UP
220/330 RESIST TERM
220/330 RESIST TERM
LED
LED
DR11-C J2 (40 PIN)
DR11-C J1 (40 PIN)

BIOMATION 8100 / DR11-C CABLE MAPPING



PRELIMINARY 9/1/83



PRELIMINARY 9/1/83

```
          PROGRAM BIO8100V2
    C     .IDENT /V2-5a/
    C
    C     Software interface between VAX and Biomation 8100
    C             transient recorder
    C
    C     This version implements full computer control of the
    C             8100s.
    C
    C     This version allows the user to disable correlation
    C             checking of the overlap region
    C
          IMPLICIT LOGICAL*4 (S)
    C
    C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C
    C
    C     I/O page mapping
    C
          PARAMETER IPROT = '0080'X        !SOGW:RWED prot mask
          PARAMETER IO_SPACE_BASE = 'FFE000'X
          PARAMETER IPGCNT = 1
          PARAMETER IUNIBUS_BASE = '760000'O
          PARAMETER IDR11_CSR = '767770'O
          PARAMETER IBUFSIZ = 8192
          INCLUDE 'BIOINSTR.INC/LIST'
  1 C
  1 C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C
  1 C
  1 C     Biomation 8100 digital control word format:
  1 C
  1 C 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
  1 C UNIT ADDR   INSTR TYP    INST     ..........DATA...............
  1 C
  1 C
  1 C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C
  1 C
  1 C     Biomation 8100 control instructions
  1 C
  1 C     Instruction types:
  1 C
  1       PARAMETER IT_CONTROL = '0800'X
  1       PARAMETER IT_OMODE   = '0C00'X
  1       PARAMETER IT_CHANA   = '1000'X
  1       PARAMETER IT_CHANB   = '1400'X
  1       PARAMETER IT_ARM     = '1800'X
  1       PARAMETER IT_TRIGGER = '1C00'X
  1 C
  1 C     Instructions.
  1 C
  1       PARAMETER I_I0       = '0000'X
  1       PARAMETER I_I1       = '0100'X
  1       PARAMETER I_I2       = '0200'X
  1       PARAMETER I_I3       = '0300'X
  1 C
  1 C     Data bit positions.
  1 C
  1       PARAMETER IBIT_0     = '0001'X
```

```
 1         PARAMETER IBIT_1     = '0002'X
 1         PARAMETER IBIT_2     = '0004'X
 1         PARAMETER IBIT_3     = '0008'X
 1         PARAMETER IBIT_4     = '0010'X
 1         PARAMETER IBIT_5     = '0020'X
 1         PARAMETER IBIT_6     = '0040'X
 1         PARAMETER IBIT_7     = '0080'X
 1         PARAMETER IBIT_A     = '00FF'X
 1 C
 1 C       Biomation 8100 instruction summary
 1 C
 1 C       B0-B7 are output bit masks
 1 C       D0-D7 are input bit masks
 1 C
 1 C C C   CONTROL FUNCTIONS - IT_CONTROL
 1 C       (2)
 1 C
 1 C       I_I0 - Set program mask
 1 C
 1 C               B7 - Trigger control group
 1 C               B6 - Arm control group
 1 C               B5 - Channel B control group
 1 C               B4 - Channel A control group
 1 C               B3 - Time base, output mode, record mode groups
 1 C               B2-B0 NOP
 1 C
 1 C       I_I1 - Set function
 1 C
 1 C               NOTE!  Only ONE bit may be set per instruction
 1 C                       load.
 1 C
 1 C               B7 - Clear and update "status" word
 1 C               B6 - Plot
 1 C               B5 - NOP
 1 C               B4 - Arm
 1 C               B3 - Trigger
 1 C               B2 - NOP
 1 C               B1 - Switch to alternate time base
 1 C               B0 - Reset
 1 C
 1 C       I_I2 - Read status
 1 C
 1 C               D7 - Record in progress
 1 C               D6 - Output data ready
 1 C               D5 - Ready to Arm
 1 C               D4 - Ready to Trigger
 1 C               D3 - Offscale Chan A (+)
 1 C               D2 - Offscale Chan A (-)
 1 C               D1 - Offscale Chan B (+)
 1 C               D0 - Offscale Chan B (-)
 1 C
 1 C       I_I3 - Enable output data
 1 C
 1 C               D7-D0 output data
 1 C
 1 C C C   OUTPUT MODE, TIME BASE, RECORD MODE - IT_OMODE
 1 C       (3)
```

```
1 C
1 C          I_I0 - Main time base
1 C
1 C                  B7        B6        SOURCE
1 C                  0         0         Internal
1 C                  0         1         External <.25 ms
1 C                  1         0         External >1 ms
1 C                  1         1         External .25 ms - 1 ms
1 C
1 C                  B5        B4        UNITS
1 C                  0         0         us
1 C                  0         1         ms
1 C                  1         0         s
1 C                  1         1         NOP
1 C
1 C          B3-B0   HEX       RANGE
1 C                  0         .01
1 C                  1         .02
1 C                  2         .05
1 C                  3         .1
1 C                  4         .2
1 C                  5         .5
1 C                  6         1.
1 C                  7         2.
1 C                  8         5.
1 C                  9         10.
1 C                A-F         UNDEFINED
1 C
1 C          I_I1 - Alternate time base
1 C
1 C                  Identical to Main Time Base
1 C
1 C          I_I2 - Output mode
1 C
1 C                  B7-B2 NOP
1 C
1 C                  B1        B0        MODE
1 C                   0         0        OFF
1 C                   0         1        AUTO
1 C                   1         0        EDIT
1 C                   1         1        UNDEFINED
1 C
1 C          I_I3 - Record mode
1 C
1 C                  B7-B6     NOP
1 C
1 C                  B5        B4        START RECORD
1 C                  0         0         Arm
1 C                  0         1         Delayed Arm
1 C                  1         0         Trigger
1 C                  1         1         Delayed Trigger
1 C
1 C                  B3        B2        SWITCH TO ALT TIMEBASE
1 C                  0         0         NOP
1 C                  0         1         Delayed Arm
1 C                  1         0         Trigger
1 C                  1         1         Delayed Trigger
```

```
1 C
1 C                  B1       BO       STOP RECORD
1 C                  0        0        End-of-sweep
1 C                  0        1        NOP
1 C                  1        0        Trigger
1 C                  1        1        Delayed Trigger
1 C
1 C C C   CHANNEL A FUNCTIONS - IT_CHANA
1 C       (4)
1 C
1 C       I_I0 - Range and coupling
1 C
1 C                  B7       B6       + INPUT COUPLING
1 C                  0        0        Off
1 C                  0        1        AC
1 C                  1        0        DC
1 C                  1        1        NOP
1 C
1 C                  B5       B4       - INPUT COUPLING
1 C                  0        0        Off
1 C                  0        1        AC
1 C                  1        0        DC
1 C                  1        1        NOP
1 C
1 C       B3-B0      HEX      RANGE
1 C                  0        +- .05 (V)
1 C                  1        +- .1
1 C                  2        +- .2
1 C                  3        +- .5
1 C                  4        +- 1.
1 C                  5        +- 2.
1 C                  6        +- 5.
1 C                7-F        UNDEFINED
1 C
1 C       I_I1 - Input offset magnitude
1 C
1 C                  BCD coded in lower byte
1 C       ex:
1 C                  value 99 = '99'X
1 C
1 C       I_I2 - Input mode and offset sign
1 C
1 C                  B7-B5    NOP
1 C
1 C                  B4       MODE
1 C                  0        Off
1 C                  1        Input
1 C
1 C                  B3-B1    NOP
1 C
1 C                  B0       POLARITY
1 C                  0        -
1 C                  1        +
1 C
1 C       I_I3 - NOP
1 C
1 C C C   CHANNEL B FUNCTIONS - IT_CHANB
```

```
1 C         (5)
1 C
1 C         Identical to Channel A functions
1 C
1 C         I_I0 - Range and coupling
1 C         I_I1 - Input offset magnitude
1 C         I_I2 - Input mode and offset sign
1 C         I_I3 - NOP
1 C
1 C C C     ARM FUNCTIONS - IT_ARM
1 C         (6)
1 C
1 C         I_I0 - Delay magnitude, 2nd and least digits
1 C
1 C                 BCD coded in lower byte
1 C
1 C         I_I1 - Delay magnitude, most significant digit
1 C
1 C                 BCD coded in lowest 4 bits of lower byte
1 C
1 C         I_I2 - Level magnitude
1 C
1 C                 BCD coded in lower byte
1 C
1 C         I_I3 - Arm mode, source, coupling, slope and level
1 C                 polarity
1 C
1 C                 B7      MODE
1 C                 0       Input
1 C                 1       Auto
1 C
1 C                 B6      NOP
1 C
1 C                 B5      SOURCE
1 C                 0       External
1 C                 1       Internal
1 C
1 C                 B4      Internal Source
1 C                 0       Chan A
1 C                 1       Chan B
1 C
1 C                 B3      SLOPE
1 C                 0       +
1 C                 1       -
1 C
1 C                 B2      COUPLING
1 C                 0       DC
1 C                 1       AC
1 C
1 C                 B1      NOP
1 C
1 C                 B0      LEVEL POLARITY
1 C                 0       -
1 C                 1       +
1 C
1 C C C     TRIGGER FUNCTIONS - IT_TRIGGER
1 C         (7)
```

```
1 C
1 C      Identical to Arm Functions
1 C
1 C      I_I0 - Delay magnitude, 2nd and least digits
1 C      I_I1 - Delay magnitude, most significant digit
1 C      I_I2 - Level magnitude
1 C      I_I3 - Trigger mode, source, coupling, slope and level
1 C             polarity
1 C
  C
  C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C
  C
  C      Data arrays
  C
         INTEGER*2 TESTDATA(10240)
         INTEGER*4 TBUF(128,1)
         EQUIVALENCE(TESTDATA,TBUF)
  C
  C      Data array
  C
         INTEGER*2 IDATA(10240)
         INTEGER*4 OBUF(128,1)
         BYTE BBUF(2)
         EQUIVALENCE (IDATA,OBUF,BBUF)
         REAL*4 XVALUES(IBUFSIZ)
         REAL*4 YVALUES(IBUFSIZ)
         INTEGER*4 WSETLM
  C
  C      Skeleton ILS header
  C
         INTEGER*4 ILSHDR(128)
  C
  C      Misc. variables
  C
         INTEGER*4 RTPRIOR
         INTEGER*2 IVAL
         BYTE BVAL
         REAL*4 SAMP
         REAL*4 SAMPINT(10) /.01,.02,.05,.1,.2,.5,1.,2.,5.,10./
         INTEGER*4 GNOSIDX(4) /0,1,2,7/
         REAL*4 OFFSET(4) /-1.0,-2.79,2.0,0.0/
         REAL*4 GAIN(4) /1.,1.,1.,1./
         INTEGER*4 ISMULT(3) /0,3,6/
         LOGICAL*4 BIO_CODE,BIO_SYNCH
         CHARACTER*80 INPUT_LINE
         CHARACTER*63 FULL_FILE
         CHARACTER*20 COMMAND
         LOGICAL*4 INVALID
         LOGICAL*4 SINIT /.FALSE./
         REAL*4 CORCO(11)
         LOGICAL*4 CORRCHK /.FALSE./
  C
         EXTERNAL SEC$M_WRT,SEC$M_PFNMAP
  C
  C      Array mapped into system space
  C
         INTEGER*2 IDUMMY
```

```
      COMMON /BIOPAG/ IDUMMY(256)
C
C     Device unit addresses in order of access
C
      INTEGER*4 UNUM(8)
      INTEGER*4 NUNITS
C
C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C
C
C     Longword arrays *2 for virtual addresses generated
C     by mapped section SGR's
C
      INTEGER*4  DATSEC(2), DATRET(2)
      INTEGER*4  LCKREQ(2), LCKRET(2)
      INTEGER*4  CODREQ(2), CODRET(2)
      INTEGER*4 VBN
      INTEGER*4 CODEST, CODEND
      CHARACTER*63 FILENAME
C
C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C
C
C     EXECUTABLE STATEMENTS START HERE
C
C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C
C
C     Obtain parameters from the user
C
C     This version supports multiple units in delayed trigger
C             mode.  This allows longer records to be
C             captured.  A maximum of 4 units can be
C             connected.
C
1234  WRITE (6,1111)
1111  FORMAT ('$Graphics output? [1=yes]: ')
      READ (5,3) IGRAF
C
1     WRITE (6,2)
2     FORMAT ('$How many Biomation 8100s in use: ')
      READ (5,3,ERR=1) NUNITS
3     FORMAT (I)
      IF (NUNITS.EQ.0) NUNITS=1
      IF (NUNITS.GT.4) THEN
              WRITE (6,5)
5             FORMAT (' MAXIMUM OF 4 UNITS 'N LONG RECORD ',
     1                'FORMAT')
              GO TO 1
      ENDIF
7     WRITE (6,8)
8     FORMAT ('$Unit address(es): ')
      READ (5,9,ERR=7) (UNUM(J),J=1,NUNITS)
9     FORMAT (I,I,I,I,I,I,I,I)
C
10    WRITE (6,15)
15    FORMAT ('$Nr of channels [1 or 2]:  ')
      READ (5,17,ERR=10,END=10) NCHANS
17    FORMAT (I)
      IF (NCHANS.LT.1.OR.NCHANS.GT.2) GO TO 10
```

```
C
20      WRITE (6,25)
25      FORMAT ('$Starting WD file number:   ')
        READ (5,17,ERR=20,END=20) IFIRST
        IF (IFIRST.GE.9999) THEN
                TYPE *,'INVALID WD FILE NUMBER:  ',IFIRST
                GO TO 20
        ENDIF
C
30      WRITE (6,35)
35      FORMAT ('$WD file number increment:  ')
        READ (5,17,ERR=30,END=30) INCR
        IF (INCR.EQ.0) INCR=1
C
40      WRITE (6,41)
41      FORMAT ('$Enter numeric portion of sample interval:  ')
        READ (5,42) INPUT_LINE
42      FORMAT (A)
        STATUS=STR$TRIM(INPUT_LINE,INPUT_LINE,INLEN)
        IF (INLEN.EQ.0) GO TO 40
        IPOS=INDEX(INPUT_LINE(1:INLEN),'.')
        IF (IPOS.EQ.0) THEN
                INLEN=INLEN+1
                INPUT_LINE(INLEN:INLEN)='.'
        ENDIF
        READ (INPUT_LINE(1:INLEN),44,ERR=40) SAMP
44      FORMAT (F)
C
C       Is this value in the list?
C
        ISNR=0
        DO I=1,10
                IF (SAMP.EQ.SAMPINT(I)) ISNR=I
        ENDDO
        IF (ISNR.EQ.0) THEN
                WRITE (6,46) SAMP
46              FORMAT (' INVALID PARAMETER:  ',F)
                GO TO 40
        ENDIF
C
50      WRITE (6,51)
51      format ('0Units:      S  - 1',/,
        1      '              mS - 2',/,
        2      '              uS - 3')
        WRITE (6,52)
52      FORMAT ('$Select [1-3]:  ')
        READ (5,17,ERR=50,END=50) IMUL
        IF (IMUL.LT.1.OR.IMUL.GT.3) GO TO 50
C
C       Biomation 8100 will not do dual channel operation
C               at a rate above 10 Mhz
C
        IF (NCHANS.EQ.2.AND.(ISNR.LT.4).AND.IMUL.EQ.3) THEN
                WRITE (6,62)
62              FORMAT (' ERROR.  TWO CHANNEL OPERATION MAX 10Mhz')
                STOP
        ENDIF
```

```
C
C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C
C
C       Determine the ILS format sampling frequency
C
        IF (ISNR.LE.7) THEN
                SAMP=1.0/SAMPINT(ISNR)
                ISAMP=SAMP
                MULT=ISMULT(IMUL)
        ELSE
                SAMP=100.0/SAMPINT(ISNR)
                ISAMP=SAMP
                MULT=ISMULT(IMUL)-2
        ENDIF
C
C       Load the skeleton ILS header
C
C       58 - Starting channel
C       59 - Number of channels
C       61 - Frequency exponent
C       62 - Frequency
C       63 - Flag for sampled data file
C
        ILSHDR(58)=1
        ILSHDR(59)=NCHANS
        ILSHDR(61)=MULT
        ILSHDR(62)=ISAMP
        ILSHDR(63)=-32000
C
C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C
C
C       If this is a restart (BINIT=.TRUE.) then skip the
C               I/O page mapping
C
        IF (BINIT) GO TO 200
C
C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C
C
C       VBN is the virtual block number of the start of
C               UNIBUS address space
C
        VBN=IO_SPACE_BASE/'200'X
        IOS=(IDR11_CSR-IUNIBUS_BASE)/'200'X
C
C       The DR11-C device register are on page IOS of the
C               UNIBUS I/O space
C
        VBN=VBN+IOS
C
C       Obtain the process base address for the page of I/O
C               space containing the device registers
C
        DATSEC(1)=%LOC(IDUMMY)
C
C       Make sure the array used is page aligned
C
        DATSEC(2)=DATSEC(1)/'200'X
```

```
                DATSEC(2)=DATSEC(2)+'200'X
                IF (DATSEC(1).NE.DATSEC(2)) THEN
                        WRITE (6,100) DATSEC(1)
        100             FORMAT (' ERROR - ARRAY TO BE MAPPED IS NOT',
                1                       ' PAGE ALIGNED!  ADDR=',Z8.8)
                        STOP 'ERROR ABORT'
                ENDIF
        C
        C       Set flags for writable private section, using page
        C               frame mapping
        C
                SEC_FLG=%LOC(SEC$M_WRT)+%LOC(SEC$M_PFNMAP)
        C
        C       Increase the working set size as much as the system
        C               allows
        C
                MAXWSL=3072-WSETLM
                SUCCESS=SYS$ADJWSL(MAXWSL,WSETLM)
     C  D       TYPE *,MAXWSL,WSETLM
        C
        C       Perform the actual mapping of the desired page of
        C               IO space
        C
     C  d       WRITE (6,110) VBN,DATSEC(1),DATSEC(2)
        110     FORMAT (' VBN=',z4.4,2(4x,z8.8))
                STATUS=SYS$CRMPSC(DATSEC,DATRET,,%VAL(SEC_FLG),,
                1       ,,,%VAL(1),%VAL(VBN),IPROT,)
                IF (.NOT. STATUS)
                1       CALL FATAL_ERROR(STATUS,'Mapping IO page')
        C
        C       Lock the data array into memory
        C
                LCKREG(1)=%LOC(IDATA)
                LCKREG(2)=LCKREG(1)+IBUFSIZ*2-1
                STATUS=SYS$LCKPAG(LCKREG,LCKRET,)
                IF (.NOT.STATUS)
                1       CALL FATAL_ERROR(STATUS,'Locking data array')
     C  d       WRITE (6,120) LCKRET(1),LCKRET(2)
        120     FORMAT (' Data area locked: ',Z8.8,' ',Z8.8)
        C
        C       Lock the io access code into memory
        C
                STATUS=BIO_CODE(CODEST,CODEND)
                CODREG(1)=(CODEST)
                CODREG(2)=(CODEND)
                STATUS=SYS$LCKPAG(CODREG,CODRET,)
                IF (.NOT.STATUS)
                1       CALL FATAL_ERROR(STATUS,'Locking i/o code')
     C  d       WRITE (6,130) CODRET(1),CODRET(2)
        130     FORMAT (' Code area locked: ',Z8.8,' ',Z8.8)
        C
        C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C
        C
        C       Main loop:
        C
        200     CONTINUE
                SDEBUG=.FALSE.
```

```
209     WRITE (6,210)
210     FORMAT ('OMenu:  ',/,
        1         ' <c/r> - Acquire next waveform',/,
        1         ' CORr  - Toggle enable/disable correlation fitting',/,
        1         ' CLEar - Clear outputs',/,
        1         ' NEW   - Re-enter parameters',/,
        1         ' <^Z>  - Exit',/)
        WRITE (6,215)
215     FORMAT ('$Function:  ')
        READ (5,220,END=1000,ERR=200) COMMAND
220     FORMAT (A)
222     FORMAT (' ',A)
        STATUS=STR$UPCASE(COMMAND,COMMAND)
        STATUS=STR$TRIM(COMMAND,COMMAND,LCOMM)
C
        IF (LCOMM.EQ.0) GO TO 250
C
        IF (COMMAND(1:3).EQ.'NEW') THEN
                WRITE (6,222) COMMAND
                SINIT=.TRUE.
                GO TO 1234
        ENDIF
C
        IF (COMMAND(1:3).EQ.'DEB') THEN
                WRITE (6,222) COMMAND
                SDEBUG=.TRUE.
                GO TO 250
        ENDIF
C
        IF (COMMAND(1:3).EQ.'COR') THEN
                CORRCHK=.NOT.CORRCHK
                IF (CORRCHK) THEN
                        WRITE (6,222) 'Corelation ENABLED!'
                        ILBHDR(63)=-32000
                ELSE
                        WRITE (6,222) 'Correlation DISABLED!'
                        ILBHDR(63)=0
                ENDIF
                GO TO 200
        ENDIF
C
        IF (COMMAND(1:3).EQ.'CLE') THEN
                WRITE (6,222) COMMAND
                DO IVALUE=1,NUNITS
                        STATUS=UNIT_SELECT(UNUM(IVALUE))
C
C       Enable programming of output mode
C
                        INSTR=IT_CONTROL+I_IO+IBIT_3
                        CALL LOAD_CTRLW_HS(INSTR,I1,I2)
C
C       Select EDIT output mode
C
                        INSTR=IT_OMODE+I_I2+IBIT_1
                        CALL LOAD_CTRLW_HS(INSTR,I1,I2)
C
C       Return to output mode = OFF
```

```
C
                        INSTR=IT_OMODE+I_I2
                        CALL LOAD_CTRLW_HS(INSTR,I1,I2)
C
C       Return control of output mode to front panel
C
                        INSTR=IT_CONTROL+I_IO
                        CALL LOAD_CTRLW_HS(INSTR,I1,I2)
C
                ENDDO
                GO TO 200
        ENDIF
C
C       Unknown command
C
        WRITE (6,245) COMMAND
245     FORMAT (' UNKNOWN COMMAND:  ',A)
        GO TO 200
C
C       Bump process to REALTIME priority
C
250     STATUS=SYS$SETPRI(,,%VAL(31),)
        IF (.NOT. status)
        1       CALL FATAL_ERROR (status,'SETTING RT PRIO')
C
C       Perform the data transfer
C
        WRITE (6,222) 'DATA TRANSFER'
        DO IVALUE=1,NUNITS
                WRITE (6,234) UNUM(IVALUE),IVALUE,NUNITS
234             FORMAT (' Selecting unit ',1i1,' #',1i1,
        1               ' of ',1i1)
                STATUS=UNIT_SELECT(UNUM(IVALUE))
C
C       Enable programming of output mode
C
                INSTR=IT_CONTROL+I_IO+IBIT_3
                CALL LOAD_CTRLW_HS(INSTR,I1,I2)
C
C       Select EDIT output mode
C
                INSTR=IT_OMODE+I_I2+IBIT_1
                CALL LOAD_CTRLW_HS(INSTR,I1,I2)
C
C       Data transfer:
C
C       Enable digital output and transfer the digital memory
C               to the host
C
                IF (.NOT.SDEBUG) THEN
                    STATUS=BIO_SYNCH(IDATA(1+(IVALUE-1)*2048),2030)
                ELSE
                    STATUS=BIO_IN(IDATA(1+(IVALUE-1)*2048),2030)
                ENDIF
C
C       Return to output mode = OFF
C
```

```
                INSTR=IT_OMODE+I_I2
                CALL LOAD_CTRLW_HS(INSTR,I1,I2)
C
C       Return control of output mode to front panel
C
                INSTR=IT_CONTROL+I_IO
                CALL LOAD_CTRLW_HS(INSTR,I1,I2)
C
        ENDDO
C
C       Drop back to normal priority
C
        STATUS=SYS$SETPRI(,,%VAL(4),RTPRIOR)
        IF (.NOT. STATUS)
        1       CALL FATAL_ERROR (STATUS,'CLRING RT PRIORITY')
C
C       Properly sign extend the 8 bit values to 16 bits
C
300     DO I=1,IBUFSIZ
                IDX=(I*2)-1
                BVAL=BBUF(IDX)
                IDATA(I)=BVAL
        ENDDO
C
C       APPLY GAIN AND OFFSET CORRECTIONS
C
        DO J=1,NUNITS
            DO I=1,2048
                IZLCH=UNUM(J)
                DO ICRUD=1,4
                        IF (QNDSIDX(ICRUD).EQ.IZLCH) THEN
                                ICR=ICRUD
                        ENDIF
                ENDDO
                IDX=(J-1)*2048+I
                TVAL=FLOAT(IDATA(IDX))
                TVAL=(TVAL+OFFSET(ICR))*GAIN(ICR)
                IDATA(IDX)=NINT(TVAL)
            ENDDO
        ENDDO
C
C       If correlation checking is disabled, save the raw data
C
        IF (.NOT CORRCHK) THEN
                DO I=1,10240
                        TESTDATA(I)=IDATA(I)
                ENDDO
        ENDIF
C
C       Validate the two channel synchronization and merge the
C               multi-unit records
C
C       (Overlapped portions should overlay exactly)
C
        DO I=1,NUNITS-1
                M0=I*2010+1
                M1=I*2048+1
```

```
                    DO J=-5,5
                            CALL CORREL(IDATA(MO+4),IDATA(M1+J+4),10,CORCO(J+6))
                    ENDDO
C   D               WRITE (6,314) UNUM(I),UNUM(I+1)
    314             FORMAT (' ',I11,'-',I11)
    C
    C       Highest correlation will occur at best match
    C
                    VMAX=-100000.
                    IOS=0
                    DO K=-5,5
C   d                       WRITE (6,315) K,CORCO(K+6)
    315                     FORMAT (' ',I12,' ',F)
                            IF (VMAX.LT.CORCO(K+6)) THEN
                                    VMAX=CORCO(K+6)
                                    IOS=K
                            ENDIF
                    ENDDO
    C
    C       If the best match yielded a low correlation, then the data
    C               was probably not properly synched on acquisition
    C
                    IF (VMAX.LE.0.0) THEN
                            write (6,316) unum(i),unum(i+1)
    316     FORMAT (' No correlation between units',2(1x,i11))
                            GO TO 200
                    ENDIF
                    M1=M1+IOS
                    write (6,315) ios,vmax
                    WRITE (6,318) (IDATA(J),J=MO,MO+13)
                    WRITE (6,318) (IDATA(J),J=M1,M1+13)
    318             FORMAT (' ',14I5)
                    DO L=10,2024
                            IDATA(MO+L)=IDATA(M1+L)
                    ENDDO
            ENDDO
            INVALID=.FALSE.
            IF (INVALID) THEN
                    WRITE (6,320) UNUM(1),UNUM(2)
    320             FORMAT (' Channel synch lost unit ',I12,'-',I12)
                    NVALCNT=NVALCNT+1
                    GO TO 250
            ENDIF
    C
    C       Plot the entire buffer
    C
            IF (IGRAF.EQ.1) THEN
                    CALL INITT(120)
                    NUMPTS=(NUNITS*2024)/NCHANS
                    CALL BINITT
                    CALL NPTS(NUMPTS)
                    CALL XNEAT(0)
                    CALL YNEAT(0)
                    IF (NCHANS.EQ.1) THEN
                            DO I=1,NUMPTS
                                    XVALUES(I)=FLOAT(I-1)
                                    YVALUES(I)=FLOAT(IDATA(I))
```

```
                        ENDDO
                        YVALUES(1)=128.
                        YVALUES(2)=-128.
                ELSE
                        DO I=0,NUMPTS-1
                                XVALUES(I+1)=FLOAT(I)
                                YVALUES(I+1)=IDATA(2+I*2)
                        ENDDO
                        YVALUES(1)=128.
                        YVALUES(2)=-128.
                ENDIF
                CALL SLIMX(150,850)
                CALL SLIMY(245,420)
                CALL CHECK(XVALUES,YVALUES)
                CALL DSPLAY(XVALUES,YVALUES)
                IF (NCHANS.EQ.2) THEN
                        DO I=0,NUMPTS-1
                                XVALUES(I+1)=FLOAT(I)
                                YVALUES(I+1)=FLOAT(IDATA(1+I*2))
                        ENDDO
                        YVALUES(1)=128.
                        YVALUES(2)=-128.
                        CALL BINITT
                        CALL NPTS(NUMPTS)
                        CALL XNEAT(0)
                        CALL YNEAT(0)
                        CALL SLIMX(150,850)
                        CALL SLIMY(10,185)
                        CALL XLAB(0)
                        CALL CHECK(XVALUES,YVALUES)
                        CALL DSPLAY(XVALUES,YVALUES)
                ENDIF
                CALL FINITT(0,700)
        ENDIF
C
C       Open the file and write the data to disk
C
400     IF (SDEBUG) GO TO 200
        CALL BUILD_FLNM(FILENAME,IFIRST,ILEN)
        NBLKS=NUNITS*8
        OPEN (FILE=FILENAME(1:ILEN),UNIT=1,STATUS='NEW',
        1       ACCESS='DIRECT',RECL=128,ERR=500)
405     WRITE (1'1) (ILSHDR(J),J=1,128)
        IF (CORRCHK) THEN
                DO I=1,NBLKS
                        WRITE (1'I+1) (OBUF(J,I),J=1,128)
                ENDDO
        ELSE
                DO I=1,NBLKS
                        WRITE (1'I+1) (TBUF(J,I),J=1,128)
                ENDDO
        ENDIF
        INQUIRE (UNIT=1,NAME=FULL_FILE)
        STATUS=STR$TRIM(FULL_FILE,FULL_FILE,IFLEN)
        CLOSE (UNIT=1)
        WRITE (6,410) FULL_FILE(1:IFLEN)
410     FORMAT (' File generated: ',A)
```

```
              IFIRST=IFIRST+INCR
              IF (IFIRST.GT.9998) THEN
                      WRITE (6,420)
      420             FORMAT (' NO MORE WD NUMBERS AVAILABLE')
                      CALL EXIT
              ENDIF
              GO TO 200
      C
      C       Error opening desired file.
      C
      C       Two possible events:
      C       1. File already exists.  Warn the user and then
      C               open a new version (not ILS compatible)
      C       2. Any other error.  Fatal error.
      C
      500     INQUIRE (FILE=FILENAME(1:ILEN),
              1       EXIST=SEXISTS)
              IF (SEXISTS) THEN
                      WRITE (6,510) 7,FILENAME(1:ILEN)
      510             FORMAT (' ',1A1,'WARNING!  PRE-EXISTING ',
              1               'FILE ',A)
                      WRITE (6,520)
      520             FORMAT (' NEW VERSION BEING CREATED!')
                      ILEN=ILEN-1
                      OPEN (FILE=FILENAME(1:ILEN),UNIT=1,STATUS='NEW',
              1               ACCESS='DIRECT',RECL=128,ERR=500)
                      GO TO 405
              ELSE
                      STOP 'ERROR - CANNOT OPEN OUTPUT FILE'
              ENDIF
      C
      C       Normal termination
      C
      1000    IF (IGRAF.EQ.1) THEN
              ENDIF
              CALL EXIT
              END
```

```
C
C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C
C
        SUBROUTINE BUILD_FLNM(FILE,NR,LEN)
        CHARACTER*9 FILE
        INTEGER*4 RN,LEN
C
        FILE(1:9)='         '
        IF (NR.LT.10) THEN
                WRITE (FILE,10) NR
10              FORMAT ('WD',1I1,'.;1')
                LEN=6
        ENDIF
C
        IF (NR.GT.9.AND.NR.LT.100) THEN
                WRITE (FILE,20) NR
20              FORMAT ('WD',1I2,'.;1')
                LEN=7
        ENDIF
C
        IF (NR.GT.99.AND.NR.LT.1000) THEN
                WRITE (FILE,30) NR
30              FORMAT ('WD',1I3,'.;1')
                LEN=8
        ENDIF
C
        IF (NR.GT.999.AND.NR.LT.10000) THEN
                WRITE (FILE,40) NR
40              FORMAT ('WD',1I4,'.;1')
                LEN=9
        ENDIF
C
        RETURN
        END
```

```
C
C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C
C
        SUBROUTINE CORREL(IX,IY,ISIZE,COEFF)
        INTEGER*4 ISIZE
        INTEGER*2 IX(ISIZE), IY(ISIZE)
        REAL*4 COEFF
C
C       Determine the coefficient of correlation using Pearson's
C               technique
C
        REAL*4 ARRAY(5)
        REAL*4 XSUM,YSUM,X2SUM,Y2SUM,XYSUM
        EQUIVALENCE (ARRAY(1),XSUM)
        EQUIVALENCE (ARRAY(2),X2SUM)
        EQUIVALENCE (ARRAY(3),YSUM)
        EQUIVALENCE (ARRAY(4),Y2SUM)
        EQUIVALENCE (ARRAY(5),XYSUM)
C
        DO I=1,5
                ARRAY(I)=0.0
        ENDDO
C
        DO I=1,ISIZE
                X=FLOAT(IX(I))
                Y=FLOAT(IY(I))
                XSUM=XSUM+X
                YSUM=YSUM+Y
                XYSUM=XYSUM+X*Y
                X2SUM=X2SUM+X**2
                Y2SUM=Y2SUM+Y**2
        ENDDO
C
        SIZE=FLOAT(ISIZE)
C
        ZNUM=SIZE*XYSUM-XSUM*YSUM
        ZD1=SIZE*X2SUM-XSUM**2
        ZD2=SIZE*Y2SUM-YSUM**2
        ZD3=ZD1+ZD2
        IF (ZD3.LE.0.0) THEN
                COEFF=0.0
                RETURN
        ENDIF
        COEFF=ZNUM/SQRT(ZD3)
        RETURN
        END
```

```
                 0000          .TITLE BIOSUBSV2
                 0000          .IDENT /V2-3/
                 0000   ;
                 0000          $SSDEF
                 0000   ;
                 0000   ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ;
                 0000   ;
                 0000   ;      I/O subroutines for software interface between VAX and
                 0000   ;              Biomation 8100 transient recorders
                 0000   ;
                 0000   ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ;
                 0000   ;
                 0000   ;      Symbolic offsets into IO space
                 0000   ;
                 0000   ;      DR11-C CSR is at UNIBUS address 767770.  That page
                 0000   ;              begins at address 767000.  DR11 CSR is offset
                 0000   ;              by 770 (1FB hex) from the base address of the
                 0000   ;              page
                 0000   ;
000001FB         0000   DR11_CSR        =       ^X1FB
000001FA         0000   DR11_OBUF       =       DR11_CSR+2
000001FC         0000   DR11_IBUF       =       DR11_CSR+4
                 0000   ;
                 0000   ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ;
                 0000   ;
                 0000   ;      DR11-C CSR bit assignments
                 0000   ;
                 0000   ;
                 0000   ;      15 14 13 12 10 09 08 07 06 05 04 03 02 01 00
                 0000   ;
                 0000   ;      R                    R  I  I           C  C
                 0000   ;      E                    E  N  N           S  S
                 0000   ;      Q                    Q  T  T           R  R
                 0000   ;      U                    U                 1  0
                 0000   ;      E                    E  E  E
                 0000   ;      S                    S  N  N
                 0000   ;      T                    T
                 0000   ;                              A  B
                 0000   ;      B                    A
                 0000   ;
                 0000   ;
                 0000   ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ;
                 0000   ;
00008000         0000   DR11$M_REQB     =       ^X8000
00000080         0000   DR11$M_REQA     =       ^X80
00000040         0000   DR11$M_INTENA   =       ^X40
00000020         0000   DR11$M_INTENB   =       ^X20
00000002         0000   DR11$M_CSR1     =       ^X2
00000001         0000   DR11$M_CSR0     =       ^X1
00000064         0000   DELAY           =       100
                 0000   ;
                 0000   ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ;
                 0000   ;
                 0000   ;      BIOMATION 8100 CONTROL WORD FORMAT
                 0000   ;
                 0000   ;      15 14 13 12 10 09 08 07 06 05 04 03 02 01 00
                 0000   ;
```

```
                                 0000   ;       U  U  U  I  I  I  I  I  D  D  D  D  D  D  D
                                 0000   ;       N  N  N  N  N  N  N  N  A  A  A  A  A  A  A
                                 0000   ;       I  I  I  S  S  S  S  S  T  T  T  T  T  T  T
                                 0000   ;       T  T  T  T  T  T  T  T  A  A  A  A  A  A  A
                                 0000   ;                         R  R
                                 0000   ;       N  N  N  T  T  T  U  U
                                 0000   ;       R  R  R  Y  Y  Y  C  C
                                 0000   ;                P  P  P  T  T
                                 0000   ;                E  E  E  I  I
                                 0000   ;                         O  O
                                 0000   ;                         N  N
                                 0000   ;
                                 0000   ;
                                 0000   ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ;
                                 0000   ;
                                 0000   ;       SHARED GLOBAL SECTION
                                 0000   ;
                                 0000   ;       Mapped UNIBUS I/O Page containing device registers
                                 0000   ;
                            00000000            .PSECT BIOPAG PIC, GBL, SHR, NOEXE, OVR, LONG
                   00000400      0000   IOPAGE:        .BLKW           512
                                 0400   ;
                                 0400   ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ;
                                 0400   ;
                                 0400   ;       LOCAL DATA SECTION
                                 0400   ;
                            00000000            .PSECT  NONSHARED_DATA  PIC, NOEXE, LONG
                                 0000   ;
                                 0000   ;       Local variables and storage
                                 0000   ;
                           0001  0000   CSR_SET0:       .WORD   <DR11$M_CSR0>    ;!<DR11$M_CSR1>
                           0002  0002   CSR_SET1:       .WORD   DR11$M_CSR1
                           0000  0004   CSR_RESET:      .WORD   0
                           0000  0006   UNIT_NR:        .WORD   0
                           0000  0008                   .WORD   0
                           0B00  000A   OUTPUT_ENABLE:  .WORD   ^B0000101100000000
                                 000C   ;
                                 000C   ;
                                 000C   ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ;
                                 000C   ;
                            00000000            .PSECT  CODE            PIC, SHR, NOWRT, LONG
                                 0000   ;
                                 0000   ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ;
                                 0000   ;
                           0000  0000           .ENTRY BIO_CODE ^M<>
                                 0002   ;
                                 0002   ;       Return the address range of i/o access code.
                                 0002   ;       This code will be locked into memory during
                                 0002   ;               operation
                                 0002   ;
04 BC    00000016'EF      D0     0002           MOVL    CODE_START, @4(AP)
08 BC    0000001A'EF      D0     000A           MOVL    CODE_END,   @8(AP)
            50    01      D0     0012           MOVL    #SS$_NORMAL, R0
                          04     0015           RET
                                 0016   ;
                                 0016   ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ;
                                 0016   ;
```

```
                                0016   CODEST:
               00000016'        0016   CODE_START:     .ADDRESS        CODEST
               000001EB'        001A   CODE_END:       .ADDRESS        CODEND
                                001E   ;
                                001E   ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ;
                                001E   ;
                                001E   ;       Free run data input (no synch)
                                001E   ;
                                001E   ;       CALL BIO_IN(BUFFER,NPTS)
                                001E   ;
                                001E   ;       This subroutine must be preceeded by a call to
                                001E   ;       LOAD_CTRLW to put the addressed unit in digital
                                001E   ;       output mode.
                                001E   ;
                          0000  001E           .ENTRY BIO_IN ^M<>
                                0020   ;
 51    00000000'EF        DE    0020           MOVAL   IOPAGE, R1
    52     04 AC          D0    0027           MOVL    4(AP), R2
    53     08 BC          D0    002B           MOVL    @8(AP), R3
                                002F   ;
                                002F   ;       Make sure interrupts are disabled
                                002F   ;
 50    00000004'EF        B0    002F           MOVW    CSR_RESET, R0
   01F8 C1     50         B0    0036           MOVW    R0, DR11_CSR(R1)
                                003B   ;
                                003B   ;       Go through an acquisition cycle, but don't wait for
                                003B   ;               response
                                003B   ;
                                003B   100$:
 50    00000000'EF        B0    003B           MOVW    CSR_SET0, R0
   01F8 C1     50         B0    0042           MOVW    R0, DR11_CSR(R1)
 50    00000004'EF        B0    0047           MOVW    CSR_RESET, R0
   01F8 C1     50         B0    004E           MOVW    R0, DR11_CSR(R1)
 50    00000002'EF        B0    0053           MOVW    CSR_SET1, R0
   01F8 C1     50         B0    005A           MOVW    R0, DR11_CSR(R1)
   55     01FC C1         B0    005F           MOVW    DR11_IBUF(R1),R5
          82   55         B0    0064           MOVW    R5, (R2)+
 50    00000004'EF        B0    0067           MOVW    CSR_RESET, R0
   01F8 C1     50         B0    006E           MOVW    R0, DR11_CSR(R1)
             C5 53        F4    0073           SOBGEQ  R3, 100$
                                0076   ;
                                0076   ;       Make sure CSR is cleared
                                0076   ;
 50    00000004'EF        B0    0076           MOVW    CSR_RESET, R0
   01F8 C1     50         B0    007D           MOVW    R0, DR11_CSR(R1)
                                0082   ;
                                0082   ;       Address nonexistant unit addres
                                0082   ;
   50    FFFF 8F          B0    0082           MOVW    #-1, R0
   01FA C1     50         B0    0087           MOVW    R0, DR11_OBUF(R1)
         50    01         B0    008C           MOVW    #SS$_NORMAL, R0
                          04    008F           RET
                                0090   ;
                                0090   ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ;
                                0090   ;
                                0090   ;       Select the desired Biomation unit number
                                0090   ;
```

```
                  0000  0090   172          .ENTRY UNIT_SELECT ^M<>
                        0092   173 ;
      50   04 BC   D0   0092   174          MOVL    @4(AP), R0
    50   50   0D   78   0096   175          ASHL    #13, R0, R0
  00000006'EF   50   B0   009A   176          MOVW    R0, UNIT_NR
         50   01   B0   00A1   177          MOVW    #SS$_NORMAL, R0
              04   00A4   178          RET
                        00A5   179 ;
                        00A5   180 ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ;
                        00A5   181 ;
                        00A5   182 ;        Load a single control word and wait for response
                        00A5   183 ;
                        00A5   184 ;        CALL LOAD_CTRLW_HS(IWORDIN,IWORDOUT,IDUMMY)
                        00A5   185 ;                IWORDIN is the command (less address field) to
                        00A5   186 ;                        be loaded
                        00A5   187 ;                IWORDOUT is the actual command (including address)
                        00A5   188 ;                        that was loaded
                        00A5   189 ;                IDUMMY is the timeout count (not normally used)
                        00A5   190 ;
                  0000  00A5   191          .ENTRY LOAD_CTRLW_HS ^M<>
                        00A7   192 ;
51   00000000'EF   DE   00A7   193          MOVAL   IOPAGE, R1
      50   04 BC   D0   00AE   194          MOVL    @4(AP), R0
52   00000064 8F   D0   00B2   195          MOVL    #DELAY, R2
                        00B9   196 ;
                        00B9   197 ;        "OR" in the unit address to each command word
                        00B9   198 ;
50   00000006'EF   AB   00B9   199          BISW    UNIT_NR, R0
                        00C0   200 ;
                        00C0   201 ;        Return the actual control word to the calling program
                        00C0   202 ;
       08 BC   50   3C   00C0   203          MOVZWL  R0, @8(AP)
                        00C4   204 ;
                        00C4   205 ;        Load the command word
                        00C4   206 ;
     01FA C1   50   B0   00C4   207          MOVW    R0, DR11_OBUF(R1)
                        00C9   208 ;
                        00C9   209 ;        Clear REQB flipflop by pulsing CSR0
                        00C9   210 ;
50   00000000'EF   B0   00C9   211          MOVW    CSR_SET0, R0
     01FB C1   50   B0   00D0   212          MOVW    R0, DR11_CSR(R1)
50   00000004'EF   B0   00D5   213          MOVW    CSR_RESET, R0
     01FB C1   50   B0   00DC   214          MOVW    R0, DR11_CSR(R1)
                        00E1   215 ;
                        00E1   216 ;        Raise CMD signal (CSR1)
                        00E1   217 ;
50   00000002'EF   B0   00E1   218          MOVW    CSR_SET1, R0
     01FB C1   50   B0   00E8   219          MOVW    R0, DR11_CSR(R1)
                        00ED   220 ;
                        00ED   221 ;        Wait for REQUEST B
                        00ED   222 ;
                        00ED   223 120$:
     56   01FB C1   B0   00ED   224          MOVW    DR11_CSR(R1), R6
     56   8000 8F   B3   00F2   225          BITW    #DR11$M_REQB, R6
              03   12   00F7   226          BNEQU   130$
          F1 52   F5   00F9   227          SOBGTR  R2, 120$
                        00FC   228 130$:
```

```
                             00FC  ;
                             00FC  ;      Reset CMD bit
                             00FC  ;
 50  00000004'EF      B0     00FC         MOVW    CSR_RESET, R0
     01FB C1   50     B0     0103         MOVW    R0, DR11_CSR(R1)
                             0108  ;
        0C BC  52     D0     0108         MOVL    R2, @12(AP)
           50  01     B0     010C         MOVW    #SS$_NORMAL, R0
                      04     010F         RET
                             0110  ;
                             0110  ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ;
                             0110  ;
                             0110  ;      Load a control word into the command buffer
                             0110  ;
                             0110  ;      CALL LOAD_CTRLW(IWORDIN, IWORDOUT, IDUMMY)
                             0110  ;              IWORDIN is the command (less address field) to
                             0110  ;                      be loaded
                             0110  ;              IWORDOUT is the actual command (including address)
                             0110  ;                      that was loaded
                             0110  ;              IDUMMY not used
                             0110  ;
                    0000     0110         .ENTRY LOAD_CTRLW ^M<>
                             0112  ;
 51  00000000'EF      DE     0112         MOVAL   IOPAGE, R1
     50   04 BC       D0     0119         MOVL    @4(AP), R0
 52  00000064 8F      D0     011D         MOVL    #DELAY, R2
                             0124  ;
                             0124  ;      "OR" in the unit address to the command word
                             0124  ;
 50  00000006'EF      A8     0124         BISW    UNIT_NR, R0
                             0128  ;
                             0128  ;      Return the actual control word to the calling program
                             0128  ;
        08 BC  50     3C     0128         MOVZWL  R0, @8(AP)
                             012F  ;
                             012F  ;      Load the command word
                             012F  ;
     01FA C1   50     B0     012F         MOVW    R0, DR11_OBUF(R1)
           50  01     B0     0134         MOVW    #SS$_NORMAL, R0
                      04     0137         RET
                             0138  ;
                             0138  ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ;
                             0138  ;
                             0138  ;      S.W. synched data input
                             0138  ;
                             0138  ;      CALL BIO_SYNCH(BUFFER, NPTS)
                             0138  ;
                             0138  ;      Prior to calling this subroutine, the proper unit must
                             0138  ;              be selected with UNIT_SELECT, and the selected
                             0138  ;              unit must be put into EDIT MODE, DIGITAL OUTPUT
                             0138  ;              ENABLED with the proper series of LOAD_CTRLW_HS
                             0138  ;              calls
                             0138  ;
                    0000     0138         .ENTRY BIO_SYNCH ^M<>
                             013A  ;
 51  00000000'EF      DE     013A         MOVAL   IOPAGE, R1
     52   04 AC       D0     0141         MOVL    4(AP), R2
```

```
      53   08 BC   D0 0145                MOVL    @8(AP), R3
                      0149          ;
                      0149          ;     Enable digital output
                      0149          ;
  50   0000000A'EF  B0 0149                MOVW    OUTPUT_ENABLE, R0
                      0150          ;
                      0150          ;     "OR" in the unit address to each command word
                      0150          ;
  50   00000006'EF  A8 0150                BISW    UNIT_NR, R0
                      0157          ;
                      0157          ;     Load the command word
                      0157          ;
     01FA C1    50  B0 0157                MOVW    R0, DR11_OBUF(R1)
                      015C          ;
                      015C          ;     Selected unit should be in digital output mode
                      015C          ;             with the first data value ready.  Check the
                      015C          ;             status of the device
                      015C          ;
                      015C          ;     Check "READY" bit in CSR (actually FLG signal)
                      015C          ;
     55   01FB C1   B0 015C                MOVW    DR11_CSR(R1),R5
     55   0080 8F   B3 0161                BITW    #DR11$M_REGA, R5
      0000018A'EF   17 0166                jmp     300$
               1C   12 016C                BNEQU   300$
                      016E          ;
                      016E          ;     Device not ready
                      016E          ;     Select unit 7 (not normally used) and clear CSR
                      016E          ;
                      016E      290$:
  50   00000004'EF  B0 016E                MOVW    CSR_RESET, R0
     01FB C1    50  B0 0175                MOVW    R0, DR11_CSR(R1)
     50   FFFF 8F   B0 017A                MOVW    #-1, R0
     01FA C1    50  B0 017F                MOVW    R0, DR11_OBUF(R1)
     50   20D4 8F   B0 0184                MOVW    #SS$_DEVINACT, R0
                    04 0189                RET
                      018A      300$:
                      018A          ;
                      018A          ;     Transfer the first data value to the user buffer
                      018A          ;
                      018A          ;     MOVW    DR11_IBUF(R1), R5
                      018A          ;     MOVW    R5, (R2)+
                      018A          ;
                      018A          ;     Clear REQB flipflop by pulsing CSR0
                      018A          ;     Raise CMD signal (CSR1)
                      018A          ;
                      018A      310$:
  50   00000000'EF  B0 018A                MOVW    CSR_SET0, R0
     01FB C1    50  B0 0191                MOVW    R0, DR11_CSR(R1)
  50   00000004'EF  B0 0196                MOVW    CSR_RESET, R0
     01FB C1    50  B0 019D                MOVW    R0, DR11_CSR(R1)
  50   00000002'EF  B0 01A2                MOVW    CSR_SET1, R0
     01FB C1    50  B0 01A9                MOVW    R0, DR11_CSR(R1)
                      01AE          ;
                      01AE          ;     Wait for REQUEST B
                      01AE          ;
                      01AE      320$:
  50   00000064 8F  D0 01AE                MOVL    #DELAY, R0
```

```
                          01B5    343 321$:
          FD 50      F5   01B5    344            SOBGTR  R0, 321$
     56   01FB C1    B0   01B8    345            MOVW    DR11_CSR(R1), R6
     56   8000 8F    B3   01BD    346            BITW    #DR11$M_REQB, R6
               EA    13   01C2    347            BEQLU   320$
                          01C4    348 ;
                          01C4    349 ;          BIO 8100 has responded.  Store data value.
                          01C4    350 ;
     55   01FC C1    B0   01C4    351            MOVW    DR11_IBUF(R1),R5
          82    55   B0   01C9    352            MOVW    R5, (R2)+
                          01CC    353 ;
                          01CC    354 ;          Reset CMD bit
                          01CC    355 ;
50   00000004'EF     B0   01CC    356            MOVW    CSR_RESET, R0
   01FB C1      50   B0   01D3    357            MOVW    R0, DR11_CSR(R1)
          AF 53      F4   01D8    358            SOBGEQ  R3, 310$
                          01DB    359 ;
                          01DB    360 ;          Clear CSR
                          01DB    361 ;
50   00000004'EF     B0   01DB    362            MOVW    CSR_RESET, R0
   01FB C1      50   B0   01E2    363            MOVW    R0, DR11_CSR(R1)
          50    01   B0   01E7    364            MOVW    #SS$_NORMAL, R0
                     04   01EA    365            RET
                          01EB    366 ;
                          01EB    367 CODEND:
                          01EB    368 ;
                          01EB    369            .END
```

Appendix B

ILS SOFTWARE MODULES

Appendix B

ILS SOFTWARE MODULES

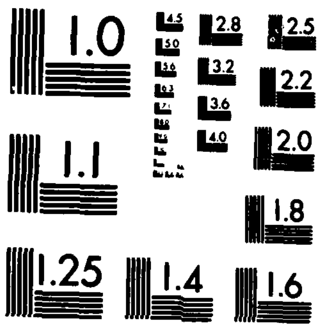
The ILS Interactive Laboratory System™ interactive digital signal processing software package (available through software leasing agreements from Signal Technology Inc., Santa Barbara, California) was used as the primary software tool in developing the signal processing software used in this study. The basic ILS package supplies a very flexible and powerful set of modular software components, implemented in an interactive environment featuring considerable graphics feedback. The ILS software provides many generic digital signal processing modules useful in a wide variety of application areas. When we found that ILS did not provide the necessary functionality needed in the acoustic emission application, new ILS modules were written, or modifications were made to existing ILS modules to implement the new functionality. These software modules and documentation are included on the following pages of this Appendix.

™ Registered Trademark

AD-A150 170 EMPLOYMENT OF ADAPTIVE LEARNING TECHNIQUES FOR THE DISCRIMINATION OF ACOU..(U) GENERAL ELECTRIC CORPORATE RESEARCH AND DEVELOPMENT SCHENECTA.. J W ERKES ET AL. 2/2

UNCLASSIFIED DEC 84 84SRD002 N00014-82-C-2031 F/G 20/1 NL

END



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

Existing ILS commands as supplied by STI:

ADF	ARITHMETIC FUNCTIONS ON INTEGER DATA
AFP	3-D AREA FUNCTION PLOT OF ANALYSIS DATA
ANL	LINEAR PREDICTION ANALYSIS USING AUTOCORRELATION, COVARIANCE, OR BURG'S METHOD
API	INVERSE FILTER ANALYSIS USING AUTOCORRELATION METHOD WITH PITCH DETECTION USING CEPSTRAL METHOD
ASG	ASSIGN LOGICAL UNIT NUMBER FOR KEYBOARD
AVG	RECORD AVERAGING (MOVING OR EXPONENTIAL) INPUT IS PRIMARY FILE, OUTPUT SECONDARY
BOP	BINARY ARITHMETIC OPERATIONS ON FLOATING POINT DATA; OR COMBINE TWO REAL VECTORS INTO ONE COMPLEX VECTOR - TWO FILES INPUT, ONE OUTPUT
BPA	PATTERN MATCHING OF TEST AND REFERENCE DATA SUPPORTS DYNAMIC PROGRAMMING FOR APPLICATIONS SUCH AS WORD RECOGNITION
CEP	CEPSTRUM DISPLAY OF VARIABLE LENGTH FRAMES
CLA	CURSOR LABELING OF PERIODIC SAMPLED DATA
CNV	CONVOLUTION OF A LONG DURATION TIME SERIES WITH A FILTER IMPULSE RESPONSE USING SHORT OVERLAPPED FFT'S
COR	CORRELATION OF TWO LONG DURATION TIME SERIES USING SHORT OVERLAPPED FFT'S
CST	COMPUTATION OF SAMPLED DATA STATISTICS; PRINTS LOCATION OF PEAK VALUE
CTX	EXAMINE OR CHANGE CONTEXT, *WHICH IS THE NUMBER* OF SAMPLED DATA POINTS PER FRAME OF DATA
CUR	X-Y CURSOR, USED IN CONJUNCTION WITH DISPLAY
DPM	DISPLAY OF ANALYSIS DATA; SUPPORTS VARIABLE FRAME SIZES
DRE	DISPLAY REAL OR COMPLEX VECTORS FROM RECORD FILES
DSP	DISPLAY OF TIME SERIES; GRIDS AND FRAME LOCATIONS
EFI	ELLIPTIC, BUTTERWORTH AND CHEBYCHEV FILTER DESIGN
FDI	FREQUENCY SPECTRUM DISPLAY, FROM FFT ON SAMPLED DATA (MAXIMUM OF 512 POINTS)
FFT	FAST FOURIER TRANSFORM OF A TIME SERIES; SUPPORTS CIRCULAR SHIFTS OF THE DATA
FIL	SPECIFY, CREATE, DELETE BINARY DATA FILES
FLT	CONVOLUTION OF A TIME SERIES WITH DIGITAL FILTERS; SUPPORTS INTEGER OR FLOATING POINT DATA FILES
FPL	FREQUENCY PLOT OF SMOOTH SPECTRA FROM ANALYSIS FILE
FTR	FORMANT TRACKING OF SPEECH DATA; STORES BANDWIDTH INFORMATION AND SUPPORTS VARIABLE FRAME SIZES
GRD	PLOT A GRID FOR FREQUENCY OR AN AXIS FOR TIME
HHH	HELP WITH ILS COMMANDS
HIS	READ DATA FROM RECORD FILES AND PLOT A HISTOGRAM ON THE SCREEN
IDC	ENTER DATA INTO COMMON BLOCK

IFL	IDEAL FILTER DESIGN; SUPPORTS CIRCULAR SHIFT OF IMPULSE RESPONSE
ILS	CREATE AND INITIALIZE USER COMMON FILE
INA	INITIALIZATION OF SAMPLED DATA FILE HEADER; SUPPORTS SPECIFICATION OF LOGICAL END OF FILE WHICH IS USED IF STARTING FRAME =0 AND NUMBER OF FRAMES=1 IN SUBSEQUENT PROGRAMS
LBA	LABEL A SAMPLED DATA SEGMENT
LBF	POINT TO A LABEL FILE
LCM	LIST ILS COMMON (ALL OR PART)
LFI	DESIGNS LINEAR PHASE FINITE IMPULSE RESPONSE (FIR) FILTERS USING THE REMEX EXCHANGE ALGORITHM
LLA	LISTS LABEL FILE
LRE	LISTING OF FLOATING POINT DATA; INDEXES TIME SERIES IN SECONDS AND SPECTRAL DATA IN HERTZ
LSN	DIGITAL-TO-ANALOG CONVERSION *NOT IMPLEMENTED*
MDF	MODIFY VALUES OF DATA POINTS IN FILES WILL MODIFY FILE HEADERS AND RECORD HEADERS
MDX	MULTIPLEXING AND DEMULTIPLEXING OF MULTI-CHANNEL SAMPLED DATA FILES
MRE	MANIPULATE RECORD FILES BY EXTRACTING RECORDS, ITEMS AND ELEMENTS
MVF	DATA MOVING WITHIN A FILE; SUPPORTS ANALYSIS FRAMES AND ZEROING OF ORIGINAL DATA
NSI	SIMULATION OF NOISY DATA BY ADDING NOISE TO A SIGNAL; GENERATES PSEUDORANDOM NOISE
OPN	ALLOCATES AND OPENS RECORD FILES
PAN	PITCH SYNCHRONOUS ANALYSIS OF SAMPLED DATA
PCO	PRINCIPAL COMPONENTS ANALYSIS
PLR	SCATTER-PLOT OF DATA ELEMENTS IN RECORD FILES
PNS	PITCH SYNCHRONOUS SYNTHESIS; INCLUDES IMPROVED ALGORITHMS WITH GLOTTAL PULSE OR GAUSSIAN NOISE EXITATION
PRT	PRINT FROM BINARY FILES (OR HEADER)
QUR	QUEUEING OF FEATURES IN TIME SERIES BASED ON LABEL INFORMATION FOR PATTERN RECOGNITION. INCLUDES GENERATION OF FEATURE MATRICES FOR APPLICATIONS SUCH AS PATTERN RECOGNITION.
RAN	RESIDUE ANALYSIS USING COEFFICIENTS FROM AUTO-CORRELATION OR COVARIANCE ANALYSIS
REC	ANALOG-TO-DIGITAL CONVERSION INTO A FILE IS NOT IMPLEMENTED EQUIVALENT HARDWARE AND SOFTWARE EXISTS FOR SOME APPLICATIONS
RSO	ROOT SOLVING FOR SPECTRAL RESONANCES; STORES REAL ROOTS FOR HIGH QUALITY SPEECH SYNTHESIS
RVR	REVERSE THE ORDER OF SAMPLED DATA POINTS
SDE	SPECTRAL-DENSITY ESTIMATION COMMAND (CROSS OR AUTO)
SDI	3-DIMENSIONAL DISPLAY OF SPECTRA; SUPPORTS INTEGER TIME SERIES, FLOATING POINT TIME SERIES OR FLOATING POINT SPECTRA INPUT FILES
SGM	SPECTROGRAM DISPLAY OF ANALYSIS DATA; SUPPORTS VARIABLE FRAME SIZES AND ERASES BEFORE DISPLAY

SIF	FUNDAMENTAL FREQUENCY EXTRACTION USING SIFT ALGORITHM
SME	CALCULATES STATISTICS OF DATA IN FEATURE RECORDS
SNS	SYNTHESIZE SAMPLED DATA FROM ANALYSIS DATA
SPL	3-DIMENSIONAL SPECTRAL PLOT OF ANALYSIS DATA; SUPPORTS VARIABLE FRAME SIZES
SRE	STORE RECORDS INTO SECONDARY FILE FROM KEYBOARD INPUT, PRIMARY SAMPLED DATA FILE OR PRIMARY ANALYSIS FILE
SSP	FREQUENCY SPECTRUM DISPLAY FROM FFT OF INVERSE FILTER COEFFICIENTS
TBL	SET UP DIRECTORY TABLE FOR ILS FILE SYSTEM
TFU	PROGRAM FOR CREATING TEST DATA IN A FILE (FILE MUST BE CREATED FIRST)
TLA	COPY LABELS MEETING A GIVEN SPECIFICATION INTO A SECONDARY LABEL FILE
TRE	TRANSFER OF FLOATING POINT DATA; SUPPORTS CHANGES IN RECORD SIZE
TRF	DATA TRANSFER BETWEEN FILES; SUPPORTS ANALYSIS FRAMES
TRM	SETTING OF TERMINAL CHARACTERISTICS; SUPPORTS NUMBER OF GRAPHIC INPUT TERMINATORS AND THE HP2648 TERMINAL IN NATIVE MODE. ALSO PRINTS SYSTEM TIME AND DATE AND WILL LABEL A PLOT. WILL CLEAR SCREEN OR PUT TERMINAL IN ALPHA MODE
TSI	TEST SIGNAL, GENERATE SAMPLED DATA, LABEL OR RECORD FILES.
TTL	TRANSFERS MARKED SECTION OF SAMPLED DATA TO THE SECONDARY FILE WITH OPTIONAL LABELING
UOP	UNARY OPERATIONS, PHASE UNWRAPPING OF FFT DATA; USES A MUCH IMPROVED ALGORITHM. MANY OTHER MANIPULATIONS OF FLOATING POINT DATA
VDI	VARIABLE DISTANCE THRESHOLD EVALUATION
VER	VERIFY HEADER BLOCK IN SAMPLED DATA OR ANALYSIS FILE
VTR	PLOT A VOCAL TRACT FROM SECONDARY ANALYSIS FILE OR USER'S COMMON
XPA	EXPAND, INTERPOLATE OR DOWNSAMPLE PRIMARY SAMPLED DATA FILE FOR HIGHER OR LOWER SAMPLING FREQUENCIES, BY INTERLEAVING ZEROS WITH THE ORIGINAL DATA OR SKIPPING DATA
XTR	READ DATA FROM PRIMARY RECORD FILE AND COMPUTE MAXIMUM OR MINIMUM VALUES

****

New ILS commands developed and used in the course of this research.

****

$ APF - MERGES GRAPHIC OUTPUT FILES..WILL ALTERNATELY MERGE INTO A USER CHOSEN FILE NAME.

$ AMP - HISTOGRAMS WITH STATISTICS FOR SAMPLED DATA FILES

```
$ CLR    PUTS AN ADM TERMINAL IN ALPHA MODE THEN
          CLEARS SCREEN AFTER GRAPHICS SESSION
$ CPF   - REASSIGNS GRAPHIC OUTPUT TO THE TERMINAL
$ GRA   - PUTS AN ADM TERMINAL IN GRAPHICS MODE
          THEN CLEARS SCREEN.............
$ CXP   - MODIFICATION OF CEP TO WRITE DATA WHICH WAS
          PREVIOUSLY PLOTTED, TO A FILE.


$ HED   - LIST CONTENTS OF THE SPS PORTION OF ILS HEADER
$ HFL   - APPEND HEADER TO YOUR DATA
$ HPF   - PROCESSES GRAF.TMP FILES RECOGNIZING FORM FEED
          NOT TO BE USED WITH OVERLAYS. DEVELOPED FOR
          USE WITH HSP.
$ HSP   - HISTOGRAM OF DATA PEAKS AND ENERGY CONTENT OF
          DATA FILES.
$ ITN   - INSERTS TEST NAME,COMMENTS AND ISPS HEADER
          VALUES INTO ISPS PORTION OF HEADER
$ LAB   - LABELS FOR PLOTS.  CREATES A GRAF.TMP FILE TO
          BE MERGED TO PROVIDE HORZ _VERT LABELING.
$ MST   - WRITES STATISTICS FROM SAMPLED DATA FILES
          AS FROM ILS COMMAND "$ CST", TO THE FILE
          "CSTAT.TMP" WITH LEGEND AND HEADINGS.
          SUBSEQUENT USES APPEND DATA.
$ OPF   - ASSIGNS GRAPHIC OUTPUT TO A FILE NAMED GRAF.TMP
$ PPF   - PROCESSES AND PRINTS GRAF.TMP FILES
$ RR    - LISTS COMMANDS WHICH USE RECORD DATA
$ RST   - WRITES STATISTICS FROM RECORD DATA FILES
          AS FROM ILS COMMAND "$ CST", TO THE FILE
          "CSTAT.TMP" WITH LEGEND AND HEADINGS.
          SUBSEQUENT USES APPEND DATA.
$ RTS   - RECORD DATA TO SAMPLED DATA FOR DISPLAY PURPOSES
$ STN   - SEARCHES A DIRECTORY FOR WD FILES FROM A
          GIVEN TEST, ALTERNATELY USES WD NUMBERS FROM
          A FILE.
$ TRM   - TRM M WILL PRINT VERTICALLY IF N1 IS NEGATIVE
$ WIN   - APPLIES FULL OR PARTIAL HANNING WINDOW
          TO SAMPLED DATA
$ WTN   - WRITES ISPS PORTION OF FILE HEADER TO
          SECONDARY FILE

$ XCP   - COMPUTES COMPLEX CEPSTRUM FROM INPUT WAVEFORM,
          (INCLUDES EXPONENTIAL WEIGHTING, AND BANDPASS
          MAPPING)
```

```
*****************************************************************
Specialized Interactive "Recipes" using ILS commands
*****************************************************************


*****************************************************************
```

```
**********************************************************************
XCPILOW.COM - an ILS "recipe" that  implements a forward complex
cepstrum, low-pass filters the cepstrum, and finally does an
inverse complex cepstrum.
**********************************************************************
```

```
$! LOW-PASS CEPSTRUM AND INVERSE TRANSFORM
$ ON CONTROLY THEN GOTO CLEANUP
$ ON ERROR THEN GOTO CLEANUP
$ IF P1 .EQS. "" THEN INQUIRE P1 "INPUT FILE NUMBER?"
$ IF P2 .EQS. "" THEN INQUIRE P2 "OUTPUT FILE NUMBER?"
$ IF P3 .EQS. "" THEN INQUIRE P3 "STARTING CHANNEL NO.?"
$ IF P4 .EQS. "" THEN INQUIRE P4 "ENDING CHANNEL NO.?"
$ IF P5 .EQS. "" THEN INQUIRE P5 "CONTEXT?"
$ CTX 'P5'
$ FIL SO
CZ9996,,1
CZ9997,,1

$ FIL 'P1'
$ FIL S9996
$ XCP Z'P3','P4'
$ FIL 9996
$ FIL S9997
$ XCP IS
$ RENAME WD9997. WD'P2'.
$CLEANUP:
$ dele/NOCONFIRM WD9996.;1
```

```
**********************************************************************
XCPIHIGH.COM - an ILS "recipe" that  implements a forward complex
cepstrum, high-pass filters the cepstrum, and finally does an
inverse complex cepstrum.
**********************************************************************
```

```
$! HIGH-PASS CEPSTRUM AND INVERSE TRANSFORM
$ ON CONTROLY THEN GOTO CLEANUP
$ ON ERROR THEN GOTO CLEANUP
$ IF P1 .EQS. "" THEN INQUIRE P1 "INPUT FILE NUMBER?"
$ IF P2 .EQS. "" THEN INQUIRE P2 "OUTPUT FILE NUMBER?"
$ IF P3 .EQS. "" THEN INQUIRE P3 "U.P. OF 1st BAND?"
$ IF P4 .EQS. "" THEN INQUIRE P4 "L.B. OF 2nd BAND?"
$ IF P5 .EQS. "" THEN INQUIRE P5 "CONTEXT?"
$ CTX 'P5'
$ FIL SO
CZ9996,,1
CZ9997,,1
CZ9998,,1

$ FIL 'P1'
$ FIL S9996
```

```
$ XCP Z1,'P3'
$ FIL 9996
$ FIL S9997
$ XCP Z'P4','P5'
$ FIL 9997
$ FIL S9998
$ XCP IS
$ RENAME WD9998. WD'P2'.
$CLEANUP:
$ dele/NOCONFIRM WD9996.;1
$ dele/NOCONFIRM WD9997.;1
```

```
*******************************************************************
XCPIPLOW.COM - an ILS "recipe" that  implements bandpass mapping
for bandpass limited signals, followed by a forward complex
cepstrum, a low-pass filtering operation on the cepstrum, and
finally an inverse complex cepstrum.
*******************************************************************
```

```
$! LOW-PASS CEPSTRUM AND INVERSE TRANSFORM
$! MODIFIED FOR BAND-PASS SYSTEM
$ ON CONTROLY THEN GOTO CLEANUP
$ ON ERROR THEN GOTO CLEANUP
$ IF P1 .EQS. "" THEN INQUIRE P1 "INPUT FILE NUMBER?"
$ IF P2 .EQS. "" THEN INQUIRE P2 "OUTPUT FILE NUMBER?"
$ IF P3 .EQS. "" THEN INQUIRE P3 "STARTING CHANNEL NO.?"
$ IF P4 .EQS. "" THEN INQUIRE P4 "ENDING CHANNEL NO.?"
$ IF P5 .EQS. "" THEN INQUIRE P5 "CONTEXT?"
$ CTX 'P5'
$ FIL SO
CZ9996,,1
CZ9997,,1

$ FIL 'P1'
$ FIL S9996
$ XCP PZ'P3','P4'
$ FIL 9996
$ FIL S9997
$ XCP ISP
$ RENAME WD9997. WD'P2'.
$CLEANUP:
$ dele/NOCONFIRM WD9996.;1
```

```
*******************************************************************
XCPIPHIGH.COM - an ILS "recipe" that  implements bandpass mapping
for bandpass limited signals, followed by a forward complex
cepstrum, a high-pass filtering operation on the cepstrum, and
finally an inverse complex cepstrum.
*******************************************************************
```

```
$! HIGH-PASS CEPSTRUM AND INVERSE TRANSFORM
$! MODIFIED FOR BAND-PASS SYSTEM
$ ON CONTROLY THEN GOTO CLEANUP
```

```
$ ON ERROR THEN GOTO CLEANUP
$ IF P1 .EQS. "" THEN INQUIRE P1 "INPUT FILE NUMBER?"
$ IF P2 .EQS. "" THEN INQUIRE P2 "OUTPUT FILE NUMBER?"
$ IF P3 .EQS. "" THEN INQUIRE P3 "U.P. OF 1st BAND?"
$ IF P4 .EQS. "" THEN INQUIRE P4 "L.B. OF 2nd BAND?"
$ IF P5 .EQS. "" THEN INQUIRE P5 "CONTEXT?"
$ CTX 'P5'
$ FIL SO
CZ9996,,1
CZ9997,,1
CZ9998,,1

$ FIL 'P1'
$ FIL S9996
$ XCP PZ1,'P3'
$ FIL 9996
$ FIL S9997
$ XCP PZ'P4','P5'
$ FIL 9997
$ FIL S9998
$ XCP ISP
$ RENAME WD9998. WD'P2'.
$CLEANUP:
$ dele/NOCONFIRM WD9996.;1
$ dele/NOCONFIRM WD9997.;1
```

```
*********************************************************************
FDECON.COM -an ILS "recipe" that implements Fourier Deconvolution
*********************************************************************
```

```
$! FOURIER DECONVOLUTION
$ ON CONTROLY THEN GOTO CLEANUP
$ ON ERROR THEN GOTO CLEANUP
$ IF P1 .EQS. "" THEN INQUIRE P1 "INPUT A-FILE NUMBER?"
$ IF P2 .EQS. "" THEN INQUIRE P2 "INPUT B-FILE NUMBER?"
$ IF P3 .EQS. "" THEN INQUIRE P3 "OUTPUT FILE NUMBER?"
$ FIL S9993
$ OPN S5
$ FIL 'P1'
$ SRE 1,1
$ FIL 'P2'
$ FIL S9994
$ SRE 1,1
$ FIL 9993
$ FIL S9995
$ FFT
$ FIL 9994
$ FIL S9996
$ FFT
$ FIL 9995
$ FIL B9996
$ FIL S9997
$ BOP D
$ FIL 9997
```

```
$ FIL S'P3'
$ OPN S1
$ FFT I
$CLEANUP:
$ dele/NOCONFIRM WD9993.;1
$ dele/NOCONFIRM WD9994.;1
$ dele/NOCONFIRM WD9995.;1
$ dele/NOCONFIRM WD9996.;1
$ dele/NOCONFIRM WD9997.;1
```

****
FDECONP.COM - sn ILS "recipe" that implements Fourier Deconvolution on bandpass mapped waveforms.
****

```
$! FOURIER DECONVOLUTION WITHIN A FREQUENCY BAND
$ IF P1 .EQS. "" THEN INQUIRE P1 "INPUT A-FILE NUMBER?"
$ IF P2 .EQS. "" THEN INQUIRE P2 "INPUT B-FILE NUMBER?"
$ IF P3 .EQS. "" THEN INQUIRE P3 "OUTPUT FILE NUMBER?"
$ IF P4 .EQS. "" THEN INQUIRE P4 "FREQUENCY LOWER LIMIT?"
$ IF P5 .EQS. "" THEN INQUIRE P5 "FREQUENCY UPPER LIMIT?"
$ ON CONTROLY THEN GOTO DELETECOMMAND
$ OPEN/WRITE OUT FDE.COM
$ WRITE OUT "$ ON CONTROLY THEN GOTO CLEANUP"
$ WRITE OUT "$ ON ERROR THEN GOTO CLEANUP"
$ WRITE OUT "$ FIL S9993"
$ WRITE OUT "$ OPN S5"
$ WRITE OUT "$ FIL ''P1'"
$ WRITE OUT "$ SRE 1,1"
$ WRITE OUT "$ FIL ''P2'"
$ WRITE OUT "$ FIL S9994"
$ WRITE OUT "$ SRE 1,1"
$ WRITE OUT "$ FIL 9993"
$ WRITE OUT "$ FIL S9995"
$ WRITE OUT "$ FFT"
$ WRITE OUT "$ FIL 9994"
$ WRITE OUT "$ FIL S9996"
$ WRITE OUT "$ FFT"
$ WRITE OUT "$ FIL 9995"
$ WRITE OUT "$ FIL B9996"
$ WRITE OUT "$ FIL S9997"
$ WRITE OUT "$ BOP DF"
$ WRITE OUT " ''P4'.,''P5'."
$ WRITE OUT "$ FIL 9997"
$ WRITE OUT "$ FIL S''P3'"
$ WRITE OUT "$ OPN S1"
$ WRITE OUT "$ FFT I"
$ WRITE OUT "$CLEANUP:"
$ WRITE OUT "$ dele/NOCONFIRM WD9993.;1"
$ WRITE OUT "$ dele/NOCONFIRM WD9994.;1"
$ WRITE OUT "$ dele/NOCONFIRM WD9995.;1"
$ WRITE OUT "$ dele/NOCONFIRM WD9996.;1"
$ WRITE OUT "$ dele/NOCONFIRM WD9997.;1"
$ CLOSE OUT
```

```
$ @FDE.COM
$DELETECOMMAND:
$ DELETE/NOCONFIRM FDE.COM;
```

****
AVERG.COM - an ILS "recipe" that implements an averaging operation ona series of candidate waveforms.
****

```
$ ON CONTROLY THEN GOTO CLEANUP
$ ON ERROR THEN GOTO CLEANUP
$ IF P1 .EQS. "" THEN INQUIRE P1 "INPUT FILE?"
$! INPUT FILE CONTAINS WD NUMBERS, ONE PER LINE, OF FILES TO BE
 AVERAGED
$ IF P2 .EQS. "" THEN INQUIRE P2 "OUTPUT FILE NUMBER?"
$ OPEN/READ IN 'P1'
$ COUNT=0
$ FIL S9995
$ OPN S3
$LOOP:
$ READ/ENDOFFILE=THATSALL IN NUM
$ COUNT=COUNT+1
$ FIL 'NUM'
$ SRE 1,1
$ GOTO LOOP
$THATSALL:
$ FIL 9995
$ FIL S9996
$ AVG 1,'COUNT'
$ FIL 9996
$ FIL S9997
$ TRE 'COUNT',1
$ FIL 9997
$ FIL S'P2'
$ RTS 1,1
$CLEANUP:
$ dele/NOCONFIRM WD9995.;1
$ dele/NOCONFIRM WD9996.;1
$ dele/NOCONFIRM WD9997.;1
$ CLOSE IN
```

****
Special Purpose ILS Software Modules created for this Study
****

****
XCP.FOR
****

```
C...           INTERACTIVE LABORATORY SYSTEM
C...
C...           ILS COMMAND PROGRAM  ** XCP **
C...
       PROGRAM XCP
C...
       IMPLICIT INTEGER (I-N)
C...
C...   START OF DOCUMENTATION
C...
C...   SPECIAL VERSION TO USE ENTIRE FILE
C...
C...
C...   COMMAND FORMAT:
C...
C...   XCP [Z,I,F][S]N1,N2
C...
C...   ALPHABETIC ARGUMENTS:
C...
C...   Z       - ZERO CEPSTRUM RECORD FROM N1 TO N2
C...   I       - INVERSE CEPSTRUM
C...   F       - CREATE TEST FILE
C...   S       - USED WITH [I], OPTIONAL PHASE SHIFT
C...   P       - PREPROCESS WITH FREQUENCY LIMIT
C...
C...   NUMERIC ARGUMENTS:
C...
C...   N1 - STARTING FRAME
C...   N2 - NUMBER OF FRAMES
C...   WITH OPTION [Z]
C...   N1 - FIRST POINT TO ZERO
C...   N2 - LAST POINT TO ZERO
C...
C...   N3 - ASK FOR ALPHA
C...   WITH OPTION [P]
C...   N4 - LOWER FREQUENCY
C...   N5 - UPPER FREQUENCY
C...
C...   END OF DOCUMENTATION
C...
       COMMON
 PI,TWOPI,THLINC,THLCON,NFFT,NDUM,NN,L,H,H1,DVTMN2,AMULT
       COMMON /CLBF/ INSFLG,LCLBF,ICLBF(40)
       COMMON /ILSA/ NBCW,NCWBK,NDPBK,NDPF,NBDP,NCWFH,
     1              KBU,KBUIN,LPU,LUGI,LUGO,NSC,CWSC,
     2              NBA2D,MIDA2D,ICTIM(4),ICDAT(6)
       COMMON /ILSB/ IA(4),N1,N2,N3,N4,N5,N6,N7,N8,N9,N10,N11,N12
       COMMON /ILSC/ IASAV(4),N1SAV,N2SAV,N3SAV,N4SAV,N5SAV,N6SAV,
     1                N7SAV,N8SAV,N9SAV,N10SAV,N11SAV,N12SAV
       COMMON /ILSE/
 IFLPA(16),LENPA,LFILPA,NFLPA,IDKPA,IDKDPA,IDPA
       COMMON /ILSF/
 IFLSA(16),LENSA,LFILSA,NFLSA,IDKSA,IDKDSA,IDSA
       COMMON /ILSH/ FS,M,MP1,MO2,N,NSPBK,NSHFT,ICON,ISF,IHAM,LAN
```

```
      COMMON /ILSI/ LRH,IEX,ISTAN,NAN,IDF(5),NP,IVL,IC
      COMMON /ILSJ/ RC(30),A(30),R(30),F(10),B(10)
C...
      COMMON /FLC1/ IFL1(16),LEN1,LFIL1,NFL1,IDK1,IDKD1,ID1
      COMMON /FLCM/
 IFLC(16),LENC,IFLALF(4),LENALF,IFLX,IFLV,LENCOM
C...
      DIMENSION IY(16384),IR(16896),IS(16896),IX(16384)
      DIMENSION Y(16384),X(16384),DUM(16384)
      DIMENSION IHR(128),IHW(128)
      DIMENSION IOPT(5),ICHSTR(5)
      EQUIVALENCE (IHR(70),SCALR),(IHW(70),SCALW)
      EQUIVALENCE (IHR(74),ACEP) ,(IHW(74),ALPHA)
C...
C
C...
      DATA SCL1/1.0/
      DATA I1,I2,I3,I4,I5,I11,I19,I30/1,2,3,4,5,11,19,30/
      DATA ICHSTR(1),ICHSTR(2),ICHSTR(3),ICHSTR(4)/2HZ ,2HI ,2HS
 ,2HF /
      DATA ICHSTR(5)/2HP /
      DATA IHW(75),IHW(76)/0,0/
      DATA IFLG,IIBW,IIBR,IOVERFLO/4*0/
      DATA LIX,LIY,LIR,LIS/2*16384,2*16896/
      DATA ICHA,ICHNP,ICHS,ICHR/2HA ,2HNP,2HS ,2HR /
      DATA SCL0/0.0/
C...
      CALL RCOMM
C...
C...
C...  INITIALIZATION DATA
      CALL AFARG(IA,I1,I4,ICHSTR,IOPT,I5)
C...
C...  CHECK OPTIONS
C...
      IF(IOPT(1).EQ.1)THEN
C...
C...  ZERO PORTION OF CEPSTRUM
            IMODE=1
      ELSE IF (IOPT(2).EQ.1)THEN
C...
C...  INVERSE CEPSTRUM
            IMODE=2
      ELSE IF (IOPT(4).EQ.1)THEN
C...
C...  CREATE TEST FILE
            IMODE=4
      ELSE
C...
C...  FORWARD CEPSTRUM OR PREPROCESS
            IMODE=0
      END IF
C...
```

```
C...      INVERSE CEPSTRUM WITH PHASE SHIFT
          IF(IMODE.EQ.2.AND.IOPT(3).EQ.1)IMODE=3
C...
C...      SET FRAMES TO DEFAULT TO WHOLE FILE
C...
          DO 10 I=1,4
          IF(IOPT(I).NE.0)GOTO 11
10        CONTINUE
C...      FIND CEPSTRUM......SET UP NUMBER OF POINTS
C...
          IF(N1.EQ.0.AND.N2.EQ.0)THEN
                  N1A=0
                  N2A=1
          ELSE
                  N1A=N1
                  N2A=N2
          END IF
          GOTO 12
C...
C...      OPTIONS [I] AND [Z] USE THE ENTIRE FILE
11        IF(IOPT(4).EQ.1)THEN
                  NPTS=2048
                  LFILSA=2048/NDPBK
                  LFRAM=NDPF
                  NFRAM=2048/NDPF
                  GOTO 260
          END IF
          N1A=0
          N2A=1
C...
12        CALL CHKFL(NFLPA,IDKPA,IFLPA,LENPA,LFILPA,IHR,ICHS,IERR)
          IF(IERR.NE.0) GO TO 310
C...
C...      GET SAMPLING FREQUENCY
C...
          ISF=IHR(62)
          IPWR=IHR(61)
          CALL SFCNV(FS,ISF,IPWR,I1)
          CALL ANCHK(NSC,N1A,N2A,N1SAV,N2SAV,NSCA,ISTAN,
          1       NAN,NDPF,NSDBK,LFILPA)
          ISTFR=N1SAV
          NFR=N2SAV
C...
C...      GET NUMBER OF POINTS.....TRUNCATE TO A POWER OF 2
C...
          CALL FTOP(N1SAV,N2SAV,NSC,NDPF,STPT,PTS,ENDPT)
          NPTS=IFIX(PTS+0.5)
          INDX=16384
20        IF(NPTS.GE.INDX)THEN
                  NPTS=INDX
                  GOTO 30
          ELSE
                  INDX=INDX/2
```

```
      END IF
      GOTO 20
C...
C...  CONVERT BACK TO FRAMES
C...
30    PTS=FLOAT(NPTS)
      CALL PTOF(STPT,PTS,NDPF,NSC,ISTFR,NFR)
      NPTX=NPTS
      NPTY=NPTS
      IST=ISTFR
      LST=ISTFR+NFR-1
C...  SET SCALE
      IF(IMODE.EQ.1)THEN
            SCALE=1.0
      ELSE IF(IMODE.EQ.0)THEN
            SCALE=1.0
      ELSE IF(IMODE.EQ.2.OR.IMODE.EQ.3)THEN
            SCALE=SCALR
      END IF
      CALL GETD(NSC,ISTFR,NPTS,NPTY,IR,LIR,IY,IIBR,
      1     IFLPA,LENPA,LFILPA)
      IF(IIBR.LT.0)GOTO 260
C...
      CALL MI2R(IY,Y,NPTS,SCALE)
200   CONTINUE
C...  ALL POINTS READ
      LFILSA=LFILPA
260   IFLGO=2
      ITYPE=-1
      CALL CHKFL(NFLSA,IDKSA,IFLSA,LENSA,LFILSA,IHW,ITYPE,IFLGO)
      ITYPE=ICHNP
      CALL SETUP(IFLSA,LENSA,LFILSA,NFRAM,LFRAM,NCWFH,IHW,ITYPE)
      IF(NFRAM.EQ.-1)GOTO 310
C...
C...
      IHW(62) = IHR(62)
      IHW(61) = IHR(61)
      IHW(63) = -32000
      IF(IOPT(5).EQ.1)THEN
            IF(IMODE.EQ.0)THEN
                  F1=FLOAT(N4)
                  F2=FLOAT(N5)
                  IHW(75)=N4
                  IHW(76)=N5
            ELSE
                  F1=FLOAT(IHR(75))
                  F2=FLOAT(IHR(76))
                  IHW(75)=IHR(75)
                  IHW(76)=IHR(76)
            END IF
      ELSE
            IHW(75)=0
            IHW(76)=0
```

```
      END IF
      IF(IMODE.EQ.1)SCALW=SCALR
C...
      NCEP=NPTS
      AMULT=ACEP
C...
C...  MAP FORWARD TRANSFORM
C...
      IF(IOPT(5).EQ.1.AND.IMODE.EQ.0)THEN
              CALL CMAP(NCEP,Y,X,FS,F1,F2,IMODE)
C     STOP
              CALL MR2R(X,Y,NCEP,I1)
      END IF
              CALL CCPC(NCEP,Y,X,IMODE)
C...
C...  MAP INVERSE???
C...
      IF(IOPT(5).EQ.1.AND.(IMODE.EQ.2.OR.IMODE.EQ.3))THEN
              CALL CMAP(NCEP,X,Y,FS,F1,F2,IMODE)
              CALL MR2R(Y,X,NCEP,I1)
      END IF
C...
C...  X IS THE OUTPUT VARIABLE
C...
C...  SCALE OUTPUT VARIABLE
C...
      IF(IMODE.EQ.1)GOTO 275
      CALL FPPIC(X,I1,NPTS,JLOC)
      SCALE=ABS(32767./X(JLOC))
      SCALW=1./SCALE
C...  NOW THAT OUTPUT SCALE IS KNOWN..WRITE HEADER
C...  SCALE IS EQUIVALENCED TO HEADER LOCATION 70
C...  WHEN THE ZERO OPTION IS USED, NO SCALING IS DONE
C...  ALSO SET ALPHA=AMULT TO STORE IN HEADER
C...
275   ALPHA=AMULT
      CALL WHEAD(IHW,IFLSA,LENSA)
C...
C...  CONVERT TO INTEGER..SCALE FOR MAXIMUM ACCURACY
C...
      CALL MR2I(X,IX,NPTS,SCALE)
      IFRAM=1
      CALL WRITD(NSC,IFRAM,ISDB,NPTS,IS,LIS,IX,IIBW,
     1        IFLSA,LENSA,LFILSA)
280   CONTINUE
C...  DUMP BUFFER
      IIBW=-IIBW
      CALL WRITD(NSC,IFRAM,ISDB,NPTS,IS,LIS,IX,IIBW,
     1        IFLSA,LENSA,LFILSA)
100   CONTINUE
999   CONTINUE
310   CALL WCOMM
      CALL EXILS
      END
```

```
*****************************************************************
List of Fortran Modules needed to compile XCP.FOR
*****************************************************************

AMODSQ  |
FFA     |
FFS     |
FFT     |
FFT842  |
MR1DF   |
ORD1    |
ORD2    |
PHADVT  |------- contained in CCEPS.FOR and CCXTRA.FOR
PHAUNW  |
PHCHCK  |
PPVPHA  |
R2TR    |
R2TX    |
R4SYN   |
R4TR    |
R4TX    |
R8SYN   |
R8TR    |
R8TX    |
RP      |
SPCVAL  |
     ----
SHIFTF

*****************************************************************
CMAP.FOR
*****************************************************************

      SUBROUTINE CMAP(NSAMP,Y,B,FS,F1,F2,MODE)
C...
C...  MAPS FILTERED DATA WITH A SELECTED FREQUENCY RANGE
C...  INTO ENTIRE SPECTRUM FOR FORWARD CEPSTRUM OR ENTIRE
C...  FREQUENCY RANGE INTO SELECTED BAND FOR INVERSE CEPSTRUM
C...
C...  ARGUMENTS:
C...          NSAMP   - NUMBER OF SAMPLES
C...          Y       - INPUT VARIABLE TIME SERIES
C...          B       - OUTPUT VARIABLE TIME SERIES
C...          FS      - SAMPLING FREQUENCY
C...          F1      - LOWER FREQUENCY BOUND
C...          F2      - UPPER FREQUENCY BOUND
C...          MODE    - IMODE FROM CEPSTRUM
C...
C****************************************************************
C...  VARIABLE LIST
C...  DELF    FREQUENCY INCREMENT
C...  DELT    TIME INCREMENT
C...  DINT    INTERPOLATING FRACTION
C...  FINT    REAL LOCATION OF LOWER FREQUENCY BOUND
```

```
C...     FMAX    MAXIMUM FREQUENCY
C...     ILOC    INTEGER LOCATION OF INTERPOLATED FREQUENCY
C...     INTF1   INTEGER LOCATION OF LOWER FREQUENCY BOUND
C...     INTF2   INTEGER LOCATION OF UPPER FREQUENCY BOUND
C...     IOFSET  ADDER TO LOCATE FREQUENCY RANGE
C...     IPOS    LOCATION OF REAL PART OF LOWER INTERPOLATION PAIR
C...     NINT    NUMBER OF FREQUENCIES TO MAP(F2-F1), DOES NOT
C                INCLUDE F1
C...     R1      REAL PART OF LOWER INTERPOLATION PAIR
C..      R2      REAL PART OF UPPER INTERPOLATION PAIR
C...     RATIO   RATIO OF MAPPED FREQUENCIES TO TOTAL FREQUENCIES
C...     VLOC    LOCATION OF INTERPOLATED FREQUENCY
C...     X1      IMAGINARY PART OF LOWER INTERPOLATION PAIR
C...     X2      IMAGINARY PART OF UPPER INTERPOLATION PAIR
C*****************************************************************
         DIMENSION Y(1),B(NSAMP+2)
         DELT=1./FS
         DELF=FS/NSAMP
         NFREQ=NSAMP/2.
         FMAX=FS/2.
C...
C...     WHERE IN THE FREQUENCY ARRAY IS THE LOWER FREQUENCY BOUND
C...
         FINT=F1/DELF
         INTF1=IFIX(FINT)
         IF(FLOAT(INTF1).LT.FINT)INTF1=INTF1+1
         INTF2=IFIX(F2/DELF)
         NINT=INTF2-INTF1
         IOFSET=2*INTF1+1
C...
C...     CONVERT TO FREQUENCY DOMAIN
C...
C
C...     SUBROUTINE FFA REPLACES THE REAL VECTOR B(K),
 (K=1,2,...,N),
C...     WITH ITS FINITE DISCRETE FOURIER TRANSFORM.  THE DC TERM IS
C...     RETURNED IN LOCATION B(1) WITH B(2) SET TO Ø.  THEREAFTER,
 THE
C...     JTH HARMONIC IS RETURNED AS A COMPLEX NUMBER STORED AS
C...     B(2*J+1) + I B(2*J+2).  NOTE THAT THE N/2 HARMONIC IS
 RETURNED
C...     IN B(N+1) WITH B(N+2) SET TO Ø.  HENCE, B MUST BE
 DIMENSIONED
C...     TO SIZE N+2.
C...     SUBROUTINE IS CALLED AS FFA (B,N) WHERE N=2**M AND B IS AN
C...     N TERM REAL ARRAY.
C...
         CALL FFA(Y,NSAMP)
C...
C...     Y NOW CONTAINS FREQUENCY VALUES
C...     MAP SELECTED RANGE INTO ARRAY B
C...
         IF(MODE.EQ.Ø)THEN
                 B(1)=Y(IOFSET)
                 B(2)=Y(IOFSET+1)
```

```
C...
                 RATIO=FLOAT(NINT)/FLOAT(NFREQ)
C        TYPE *,DELT,DELF,NFREQ,FMAX,RATIO
C        TYPE *,INTF1,INTF2,NINT,IOFSET
                 DO 100 KJ=1,NFREQ
                         VLOC=FLOAT(KJ)*RATIO
                         ILOC=IFIX(VLOC)
                         DINT=VLOC-ILOC
                         IPOS=2*ILOC+IOFSET
                         R1=Y(IPOS)
                         R2=Y(IPOS+2)
                         X1=Y(IPOS+1)
                         X2=Y(IPOS+3)
                         B(2*KJ+1)=R1+DINT*(R2-R1)
                         B(2*KJ+2)=X1+DINT*(X2-X1)
100              CONTINUE
9999     FORMAT((I8,4(4F7.0,2X))/)
C...
C...     SHIFT PHASE
C...
                 CALL SHIFTF(B,NSAMP)
C        WRITE(3,9997)
9997     FORMAT(8X,4(5X,'INPUT'9X,'SHIFTED',4X))
C        WRITE(3,9998)
9998     FORMAT('  FREQ  ',4(2('   REAL  IMAG  ')))
C        WRITE(3,9999)(J,(Y(2*J+1),Y(2*J+2),B(2*J+1),B(2*J+2),
C        1                Y(2*J+3),Y(2*J+4),B(2*J+3),B(2*J+4),
C        2                Y(2*J+5),Y(2*J+6),B(2*J+5),B(2*J+6),
C        3                Y(2*J+7),Y(2*J+8),B(2*J+7),B(2*J+8)),
C        4                J=0,NFREQ-3,4)
         ELSE
                 RATIO=FLOAT(NFREQ)/FLOAT(NINT)
                 B(1)=0.
                 B(2)=0.
                 DO 200 KJ=1,NFREQ
                         IF(KJ.LT.INTF1.OR.KJ.GT.INTF2)THEN
                                 B(2*KJ+1)=0.
                                 B(2*KJ+2)=0.
                         ELSE
                                 INDX=KJ-INTF1
                                 VLOC=RATIO*FLOAT(INDX)
                                 ILOC=IFIX(VLOC)
                                 DINT=VLOC-ILOC
                                 IPOS=2*ILOC+1
                                 R1=Y(IPOS)
                                 R2=Y(IPOS+2)
                                 X1=Y(IPOS+1)
                                 X2=Y(IPOS+3)
                                 B(2*KJ+1)=R1+DINT*(R2-R1)
                                 B(2*KJ+2)=X1+DINT*(X2-X1)
                         END IF
200              CONTINUE
C...
         END IF
```

```
C**********************************************************************
C...    SUBROUTINE FFS SYNTHESIZES THE REAL VECTOR B(K), WHERE
C...    K=1,2,...,N. THE INITIAL FOURIER COEFFICIENTS ARE PLACED IN
C...    THE B ARRAY OF SIZE N+2.  THE DC TERM IS IN B(1) WITH
C...    B(2) EQUAL TO Ø.
C...    THE JTH HARMONIC IS STORED AS B(2*J+1) + I B(2*J+2).
C...    THE N/2 HARMONIC IS IN B(N+1) WITH B(N+2) EQUAL TO Ø.
C...    THE SUBROUTINE IS CALLED AS FFS(B,N) WHERE N=2**M AND
C...    B IS THE N TERM REAL ARRAY DISCUSSED ABOVE.
C...
        CALL FFS(B,NSAMP)
C..
        RETURN
        END

*********************************************************************
                        CCEPS.FOR
*********************************************************************


C
C-------------------------------------------------------------------
-----------
C SUBROUTINE: CCEPS
C SUBROUTINE TO COMPUTE THE COMPLEX CEPSTRUM OF A SEQUENCE X(N)
C-------------------------------------------------------------------
----------
C
      SUBROUTINE CCEPS(NX,X,ISNX,ISFX,ISSUC,CX,AUX)
C
      DIMENSION X(1),CX(1),AUX(1)
      COMMON PI,TWOPI,THLINC,THLCON,NFFT,NPTS,N,L,H,H1,DVTMN2
      LOGICAL ISSUC
      NPTS=NFFT/2
      N=12
      L=2**N
      H=FLOAT(L)*FLOAT(NFFT)
      H1=PI/H
      ISSUC=.TRUE.
      ISNX=1
C
      DO 1Ø I=1,NX
        CX(I)=X(I)
        AUX(I)=FLOAT(I-1)*X(I)
   1Ø CONTINUE
      INITL=NX+1
      IEND=NFFT+2
      DO 2Ø I=INITL,IEND
        CX(I)=Ø.Ø
        AUX(I)=Ø.Ø
   2Ø CONTINUE
```

```
C
      CALL FFA(CX,NFFT)
      CALL FFA(AUX,NFFT)
C
      IF(CX(1).LT.0.0)ISNX=-1
C
      IO=-1
      DVTMN2=0.0
      IEND=NPTS+1
      DO 30 I=1,IEND
        IO=IO+2
        IE=IO+1
        AMAGSQ=AMODSQ(CX(IO),CX(IE))
        PDVT=PHADVT(CX(IO),CX(IE),AUX(IO),AUX(IE),AMAGSQ)
        AUX(IO)=AMAGSQ
        AUX(IE)=PDVT
        DVTMN2=DVTMN2+PDVT
   30 CONTINUE
      DVTMN2=(2.*DVTMN2-AUX(2)-PDVT)/FLOAT(NPTS)
C
      PPDVT=AUX(2)
      PPHASE=0.0
      PPV=PPVPHA(CX(1),CX(2),ISNX)
      CX(1)=.5*ALOG(AUX(1))
      CX(2)=0.0
      IO=1
      DO 50 I=2,IEND
        IO=IO+2
        IE=IO+1
        PDVT=AUX(IE)
        PPV=PPVPHA(CX(IO),CX(IE),ISNX)
        PHASE=PHAUNW(X,NX,ISNX,I,PPHASE,PPDVT,PPV,PDVT,ISSUC)
C
        IF(ISSUC)GO TO 40
        ISSUC=.FALSE.
        RETURN
   40   PPDVT=PDVT
        PPHASE=PHASE
        CX(IO)=.5*ALOG(AUX(IO))
        CX(IE)=PHASE
   50 CONTINUE
C
      ISFX=(ABS(PHASE/PI)+.1)
      IF(PHASE.LT.0.0)ISFX=-ISFX
      H=PHASE/FLOAT(NPTS)
      IE=0
      DO 60 I=1,IEND
        IE=IE+2
        CX(IE)=CX(IE)-H*FLOAT(I-1)
   60 CONTINUE
```

```
C
      CALL FFS(CX,NFFT)
      RETURN
      END
C
C-------------------------------------------------------------------
------------
C SUBROUTINE: SPCVAL
C SUBROUTINE TO COMPUTRE A SPECTRAL VALUE AT A FREQUENCY
C FREQ(RADIANS) FOR SEQUENCE X(N) AND N*X(N)
C-------------------------------------------------------------------
------------
C
      SUBROUTINE SPCVAL(NX,X,FREQ,XR,XI,YR,YI)
      DIMENSION X(1)
      DOUBLE PRECISION U0,U1,U2,W0,W1,W2,A,B,C,D,A1,A2,SA0,CA0
C
C
      CA0=DBLE(COS(FREQ))
      SA0=DBLE(SIN(FREQ))
      A1=2.D+0*CA0
      U1=0.D+0
      U2=U1
      W1=U1
      W2=U1
C
      DO 10 J=1,NX
        XJ=DBLE(X(J))
        U0=XJ+A1*U1-U2
        W0=DBLE(FLOAT(J-1))*XJ+A1*W1-W2
        U2=U1
        U1=U0
        W2=W1
        W1=W0
   10 CONTINUE
C
      A=U1-U2*CA0
      B=U2*SA0
      C=W1-W2*CA0
      D=W2*SA0
      A2=DBLE(FREQ*FLOAT(NX-1))
      U1=DCOS(A2)
      U2=-DSIN(A2)
      XR=SNGL(U1*A-U2*B)
      XI=SNGL(U2*A+U1*B)
      YR=SNGL(U1*C-U2*D)
      YI=SNGL(U2*C+U1*D)
      RETURN
      END
C
C-------------------------------------------------------------------
------------
C FUNCTION: PHAUNW
C PHASE UNWRAPPING BASED ON TRIBOLET'S ADAPTIVE INTEGRATION SCHEME.
```

```
C THE UNWRAPPED PHASE ESTIMATE IS RETURNED IN PHAUNW.
C-------------------------------------------------------------------
------------
C
      FUNCTION PHAUNW(X,NX,ISNX,I,PPHASE,PPDVT,PPV,PDVT,ISCONS)
C
      DIMENSION SDVT(17),SPPV(17),X(1)
      INTEGER SINDEX(17),PINDEX,SP
      LOGICAL ISCONS,FIRST
      COMMON PI,TWOPI,THLINC,THLCON,NFFT,NPTS,N,L,H,H1,DVTMN2
C
      FIRST=.TRUE.
      PINDEX=1
      SP=1
      SPPV(SP)=PPV
      SDVT(SP)=PDVT
      SINDEX(SP)=L+1
C
      GO TO 40
C
   10 PINDEX=SINDEX(SP)
      PPHASE=PHASE
      PPDVT=SDVT(SP)
      SP=SP-1
      GO TO 40
C
   20 IF((SINDEX(SP)-PINDEX).GT.1)GO TO 30
      ISCONS=.FALSE.
      PHAUNW=0.
      RETURN
C
   30 K=(SINDEX(SP)+PINDEX)/2
C
      FREQ=TWOPI*(FLOAT(I-2)*FLOAT(L)+FLOAT(K-1))/H
      CALL SPCVAL(NX,X,FREQ,XR,XI,YR,YI)
C
      SP=SP+1
      SINDEX(SP)=K
      SPPV(SP)=PPVPHA(XR,XI,ISNX)
      XMAG=AMODSQ(XR,XI)
      SDVT(SP)=PHADVT(XR,XI,YR,YI,XMAG)
C
   40 DELTA=H1*FLOAT(SINDEX(SP)-PINDEX)
      PHAINC=DELTA*(PPDVT+SDVT(SP))
C
      IF(ABS(PHAINC-DELTA*DVTMN2).GT.THLINC)GO TO 20
C
      PHASE=PPHASE+PHAINC
      CALL PHCHCK(PHASE,SPPV(SP),ISCONS)
      IF(.NOT.ISCONS)GO TO 20
C
      IF(ABS(PHASE-PPHASE).GT.PI)GO TO 20
```

```
C
      IF(SP.NE.1)GO TO 10
      PHAUNW=PHASE
      RETURN
      END
C
C-----------------------------------------------------------------------
------------
C FUNCTION: PPVPHA
C COMPUTE THE PRINCIPLE VALUE OF THE PHASE OF A SPECTRAL VALUE
C-----------------------------------------------------------------------
------------
C
      FUNCTION PPVPHA(XR,XI,ISNX)
C
      IF(ISNX.EQ.1)PPVPHA=(ATAN2((XI),(XR)))
      IF(ISNX.EQ.(-1))PPVPHA=(ATAN2(-(XI),-(XR)))
      RETURN
      END
C
C-----------------------------------------------------------------------
------------
C FUNCTION: PHADVT
C COMPUTE THE PHASE DERIVATIVE OF A SPECTRAL VALUE OF A SEQUENCE
 X(N)
C-----------------------------------------------------------------------
------------
C
      FUNCTION PHADVT(XR,XI,YR,YI,XMAG)
C

 PHADVT=-SNGL((DBLE(XR)*DBLE(YR)+DBLE(XI)*DBLE(YI))/DBLE(XMAG))
      RETURN
      END
C
C-----------------------------------------------------------------------
------------
C FUNCTION: AMODSQ
C COMPUTE THE SQUARE OF THE MODULUS OF A COMPLEX NUMBER
C-----------------------------------------------------------------------
------------
C
      FUNCTION AMODSQ(ZR,ZI)
C
      AMODSQ=SNGL(DBLE(ZR)*DBLE(ZR)+DBLE(ZI)*DBLE(ZI))
      RETURN
      END
C
C-----------------------------------------------------------------------
------------
C SUBROUTINE: PHCHCK
C SUBROUTINE TO CHECK CONSISTENCY OF A PHASE ESTIMATE
C-----------------------------------------------------------------------
------------
```

```
C
      SUBROUTINE PHCHCK(PH,PV,ISCONS)
C
      COMMON PI,TWOPI,THLINC,THLCON,NFFT,NPTS,N,L,H,H1,DVTMN2
      LOGICAL ISCONS
C
      A0=(PH-PV)/TWOPI
      A1=FLOAT(IFIX(A0))*TWOPI+PV
      A2=A1+SIGN(TWOPI,A0)
      A3=ABS(A1-PH)
      A4=ABS(A2-PH)
C
      ISCONS=.FALSE.
      IF(A3.GT.THLCON.AND.A4.GT.THLCON)RETURN
      ISCONS=.TRUE.
C
      PH=A1
      IF(A3.GT.A4)PH=A2
      RETURN
      END
C
C-----------------------------------------------------------------------
----
C SUBROUTINE:  FFA
C FAST FOURIER ANALYSIS SUBROUTINE
C-----------------------------------------------------------------------
----
C
      SUBROUTINE FFA(B, NFFT)
C
C THIS SUBROUTINE REPLACES THE REAL VECTOR B(K),  (K=1,2,...,N),
C WITH ITS FINITE DISCRETE FOURIER TRANSFORM.  THE DC TERM IS
C RETURNED IN LOCATION B(1) WITH B(2) SET TO 0.  THEREAFTER, THE
C JTH HARMONIC IS RETURNED AS A COMPLEX NUMBER STORED AS
C B(2*J+1) + I B(2*J+2).  NOTE THAT THE N/2 HARMONIC IS RETURNED
C IN B(N+1) WITH B(N+2) SET TO 0.  HENCE, B MUST BE DIMENSIONED
C TO SIZE N+2.
C SUBROUTINE IS CALLED AS FFA (B,N) WHERE N=2**M AND B IS AN
C N TERM REAL ARRAY.  A REAL-VALUED, RADIX 8  ALGORITHM IS USED
C WITH IN-PLACE REORDERING AND THE TRIG FUNCTIONS ARE COMPUTED AS
C NEEDED.
C
      DIMENSION B(2)
      COMMON /CON/ PII, P7, P7TWO, C22, S22, PI2
C
C IW IS A MACHINE DEPENDENT WRITE DEVICE NUMBER
C
C     IW = I1MACH(2)
      IW=6
C
      PII = 4.*ATAN(1.)
      PI8 = PII/8.
      P7 = 1./SQRT(2.)
```

```
      P7TWO = 2.*P7
      C22 = COS(PI8)
      S22 = SIN(PI8)
      PI2 = 2.*PII
      N = 1
      DO 10 I=1,15
        M = I
        N = N*2
        IF (N.EQ.NFFT) GO TO 20
   10 CONTINUE
      WRITE (IW,9999)
 9999 FORMAT (30H NFFT NOT A POWER OF 2 FOR FFA)
      STOP
   20 CONTINUE
      N8POW = M/3
C
C DO A RADIX 2 OR RADIX 4 ITERATION FIRST IF ONE IS REQUIRED
C
      IF (M-N8POW*3-1) 50, 40, 30
   30 NN = 4
      INT = N/NN
      CALL R4TR(INT, B(1), B(INT+1), B(2*INT+1), B(3*INT+1))
      GO TO 60
   40 NN = 2
      INT = N/NN
      CALL R2TR(INT, B(1), B(INT+1))
      GO TO 60
   50 NN = 1
C
C PERFORM RADIX 8 ITERATIONS
C
   60 IF (N8POW) 90, 90, 70
   70 DO 80 IT=1,N8POW
        NN = NN*8
        INT = N/NN
        CALL R8TR(INT, NN, B(1), B(INT+1), B(2*INT+1), B(3*INT+1),

     *      B(4*INT+1), B(5*INT+1), B(6*INT+1), B(7*INT+1), B(1),
     *      B(INT+1), B(2*INT+1), B(3*INT+1), B(4*INT+1),
 B(5*INT+1),
     *      B(6*INT+1), B(7*INT+1))
   80 CONTINUE
C
C PERFORM IN-PLACE REORDERING
C
   90 CALL ORD1(M, B)
      CALL ORD2(M, B)
      T = B(2)
      B(2) = 0.
      B(NFFT+1) = T
      B(NFFT+2) = 0.
      DO 100 I=4,NFFT,2
        B(I) = -B(I)
```

```
 100  CONTINUE
      RETURN
      END
C
C---------------------------------------------------------------------
----
C SUBROUTINE:  FFS
C FAST FOURIER SYNTHESIS SUBROUTINE
C RADIX 8-4-2
C---------------------------------------------------------------------
----
C
      SUBROUTINE FFS(B, NFFT)
C
C THIS SUBROUTINE SYNTHESIZES THE REAL VECTOR B(K), WHERE
C K=1,2,...,N. THE INITIAL FOURIER COEFFICIENTS ARE PLACED IN
C THE B ARRAY OF SIZE N+2.  THE DC TERM IS IN B(1) WITH
C B(2) EQUAL TO Ø.
C THE JTH HARMONIC IS STORED AS B(2*J+1) + I B(2*J+2).
C THE N/2 HARMONIC IS IN B(N+1) WITH B(N+2) EQUAL TO Ø.
C THE SUBROUTINE IS CALLED AS FFS(B,N) WHERE N=2**M AND
C B IS THE N TERM REAL ARRAY DISCUSSED ABOVE.
C
      DIMENSION B(2)
      COMMON /CON1/ PII, P7, P7TWO, C22, S22, PI2
C
C IW IS A MACHINE DEPENDENT WRITE DEVICE NUMBER
C
C     IW = I1MACH(2)
      IW=6
C
      PII = 4.*ATAN(1.)
      PI8 = PII/8.
      P7 = 1./SQRT(2.)
      P7TWO = 2.*P7
      C22 = COS(PI8)
      S22 = SIN(PI8)
      PI2 = 2.*PII
      N = 1
      DO 1Ø I=1,15
        M = I
        N = N*2
        IF (N.EQ.NFFT) GO TO 2Ø
  1Ø  CONTINUE
      WRITE (IW,9999)
9999  FORMAT (3ØH NFFT NOT A POWER OF 2 FOR FFS)
      STOP
  2Ø  CONTINUE
      B(2) = B(NFFT+1)
      DO 3Ø I=1,NFFT
        B(I) = B(I)/FLOAT(NFFT)
  3Ø  CONTINUE
      DO 4Ø I=4,NFFT,2
        B(I) = -B(I)
```

```
   40  CONTINUE
       N8POW = M/3
C
C REORDER THE INPUT FOURIER COEFFICIENTS
C
       CALL ORD2(M, B)
       CALL ORD1(M, B)
C
       IF (N8POW.EQ.0) GO TO 60
C
C PERFORM THE RADIX 8 ITERATIONS
C
       NN = N
       DO 50 IT=1,N8POW
         INT = N/NN
         CALL R8SYN(INT, NN, B, B(INT+1), B(2*INT+1), B(3*INT+1),
      *      B(4*INT+1), B(5*INT+1), B(6*INT+1), B(7*INT+1), B(1),
      *      B(INT+1), B(2*INT+1), B(3*INT+1), B(4*INT+1),
  B(5*INT+1),
      *      B(6*INT+1), B(7*INT+1))
         NN = NN/8
   50  CONTINUE
C
C DO A RADIX 2 OR RADIX 4 ITERATION IF ONE IS REQUIRED
C
   60  IF (M-N8POW*3-1) 90, 80, 70
   70  INT = N/4
       CALL R4SYN(INT, B(1), B(INT+1), B(2*INT+1), B(3*INT+1))
       GO TO 90
   80  INT = N/2
       CALL R2TR(INT, B(1), B(INT+1))
   90  RETURN
       END
C
C-------------------------------------------------------------------
----
C SUBROUTINE:  R2TR
C RADIX 2 ITERATION SUBROUTINE
C-------------------------------------------------------------------
----
C
C
       SUBROUTINE R2TR(INT, B0, B1)
       DIMENSION B0(2), B1(2)
       DO 10 K=1,INT
         T = B0(K) + B1(K)
         B1(K) = B0(K) - B1(K)
         B0(K) = T
   10  CONTINUE
       RETURN
       END
C
C-------------------------------------------------------------------
----
```

```
C SUBROUTINE:  R4TR
C RADIX 4 ITERATION SUBROUTINE
C-----------------------------------------------------------------
----
C
      SUBROUTINE R4TR(INT, BØ, B1, B2, B3)
      DIMENSION BØ(2), B1(2), B2(2), B3(2)
      DO 1Ø K=1,INT
        RØ = BØ(K) + B2(K)
        R1 = B1(K) + B3(K)
        B2(K) = BØ(K) - B2(K)
        B3(K) = B1(K) - B3(K)
        BØ(K) = RØ + R1
        B1(K) = RØ - R1
   1Ø CONTINUE
      RETURN
      END
C
C-----------------------------------------------------------------
----
C SUBROUTINE: R8TR
C RADIX 8 ITERATION SUBROUTINE
C-----------------------------------------------------------------
----
C
      SUBROUTINE R8TR(INT, NN, BRØ, BR1, BR2, BR3, BR4, BR5, BR6,
 BR7,
     *    BIØ, BI1, BI2, BI3, BI4, BI5, BI6, BI7)
      DIMENSION L(15), BRØ(2), BR1(2), BR2(2), BR3(2), BR4(2),
 BR5(2),
     *    BR6(2), BR7(2), BIØ(2), BI1(2), BI2(2), BI3(2), BI4(2),
     *    BI5(2), BI6(2), BI7(2)
      COMMON /CON/ PII, P7, P7TWO, C22, S22, PI2
      EQUIVALENCE (L15,L(1)), (L14,L(2)), (L13,L(3)), (L12,L(4)),
     *    (L11,L(5)), (L1Ø,L(6)), (L9,L(7)), (L8,L(8)), (L7,L(9)),
     *    (L6,L(1Ø)), (L5,L(11)), (L4,L(12)), (L3,L(13)),
 (L2,L(14)),
     *    (L1,L(15))
C
C SET UP COUNTERS SUCH THAT JTHET STEPS THROUGH THE ARGUMENTS
C OF W, JR STEPS THROUGH STARTING LOCATIONS FOR THE REAL PART OF
 THE
C INTERMEDIATE RESULTS AND JI STEPS THROUGH STARTING LOCATIONS
C OF THE IMAGINARY PART OF THE INTERMEDIATE RESULTS.
C
      L(1) = NN/8
      DO 4Ø K=2,15
        IF (L(K-1)-2) 1Ø, 2Ø, 3Ø
   1Ø   L(K-1) = 2
   2Ø   L(K) = 2
        GO TO 4Ø
   3Ø   L(K) = L(K-1)/2
   4Ø CONTINUE
      PIOVN = PII/FLOAT(NN)
```

```
      JI = 3
      JL = 2
      JR = 2
      DO 120 J1=2,L1,2
      DO 120 J2=J1,L2,L1
      DO 120 J3=J2,L3,L2
      DO 120 J4=J3,L4,L3
      DO 120 J5=J4,L5,L4
      DO 120 J6=J5,L6,L5
      DO 120 J7=J6,L7,L6
      DO 120 J8=J7,L8,L7
      DO 120 J9=J8,L9,L8
      DO 120 J10=J9,L10,L9
      DO 120 J11=J10,L11,L10
      DO 120 J12=J11,L12,L11
      DO 120 J13=J12,L13,L12
      DO 120 J14=J13,L14,L13
      DO 120 JTHET=J14,L15,L14
        TH2 = JTHET - 2
        IF (TH2) 50, 50, 90
50      DO 60 K=1,INT
          T0 = BR0(K) + BR4(K)
          T1 = BR1(K) + BR5(K)
          T2 = BR2(K) + BR6(K)
          T3 = BR3(K) + BR7(K)
          T4 = BR0(K) - BR4(K)
          T5 = BR1(K) - BR5(K)
          T6 = BR2(K) - BR6(K)
          T7 = BR3(K) - BR7(K)
          BR2(K) = T0 - T2
          BR3(K) = T1 - T3
          T0 = T0 + T2
          T1 = T1 + T3
          BR0(K) = T0 + T1
          BR1(K) = T0 - T1
          PR = P7*(T5-T7)
          PI = P7*(T5+T7)
          BR4(K) = T4 + PR
          BR7(K) = T6 + PI
          BR6(K) = T4 - PR
          BR5(K) = PI - T6
60      CONTINUE
        IF (NN-8) 120, 120, 70
70      K0 = INT*8 + 1
        KL = K0 + INT - 1
        DO 80 K=K0,KL
          PR = P7*(BI2(K)-BI6(K))
          PI = P7*(BI2(K)+BI6(K))
          TR0 = BI0(K) + PR
          TI0 = BI4(K) + PI
          TR2 = BI0(K) - PR
          TI2 = BI4(K) - PI
          PR = P7*(BI3(K)-BI7(K))
          PI = P7*(BI3(K)+BI7(K))
```

```
      TR1 = BI1(K) + PR
      TI1 = BI5(K) + PI
      TR3 = BI1(K) - PR
      TI3 = BI5(K) - PI
      PR = TR1*C22 - TI1*S22
      PI = TI1*C22 + TR1*S22
      BIØ(K) = TRØ + PR
      BI6(K) = TRØ - PR
      BI7(K) = TIØ + PI
      BI1(K) = PI - TIØ
      PR = -TR3*S22 - TI3*C22
      PI = TR3*C22 - TI3*S22
      BI2(K) = TR2 + PR
      BI4(K) = TR2 - PR
      BI5(K) = TI2 + PI
      BI3(K) = PI - TI2
8Ø    CONTINUE
      GO TO 12Ø
9Ø    ARG = TH2*PIOVN
      C1 = COS(ARG)
      S1 = SIN(ARG)
      C2 = C1**2 - S1**2
      S2 = C1*S1 + C1*S1
      C3 = C1*C2 - S1*S2
      S3 = C2*S1 + S2*C1
      C4 = C2**2 - S2**2
      S4 = C2*S2 + C2*S2
      C5 = C2*C3 - S2*S3
      S5 = C3*S2 + S3*C2
      C6 = C3**2 - S3**2
      S6 = C3*S3 + C3*S3
      C7 = C3*C4 - S3*S4
      S7 = C4*S3 + S4*C3
      INT8 = INT*8
      JØ = JR*INT8 + 1
      KØ = JI*INT8 + 1
      JLAST = JØ + INT - 1
      DO 1ØØ J=JØ,JLAST
        K = KØ + J - JØ
        TR1 = BR1(J)*C1 - BI1(K)*S1
        TI1 = BR1(J)*S1 + BI1(K)*C1
        TR2 = BR2(J)*C2 - BI2(K)*S2
        TI2 = BR2(J)*S2 + BI2(K)*C2
        TR3 = BR3(J)*C3 - BI3(K)*S3
        TI3 = BR3(J)*S3 + BI3(K)*C3
        TR4 = BR4(J)*C4 - BI4(K)*S4
        TI4 = BR4(J)*S4 + BI4(K)*C4
        TR5 = BR5(J)*C5 - BI5(K)*S5
        TI5 = BR5(J)*S5 + BI5(K)*C5
        TR6 = BR6(J)*C6 - BI6(K)*S6
        TI6 = BR6(J)*S6 + BI6(K)*C6
        TR7 = BR7(J)*C7 - BI7(K)*S7
        TI7 = BR7(J)*S7 + BI7(K)*C7
```

```
C
      T0 = BR0(J) + TR4
      T1 = BI0(K) + TI4
      TR4 = BR0(J) - TR4
      TI4 = BI0(K) - TI4
      T2 = TR1 + TR5
      T3 = TI1 + TI5
      TR5 = TR1 - TR5
      TI5 = TI1 - TI5
      T4 = TR2 + TR6
      T5 = TI2 + TI6
      TR6 = TR2 - TR6
      TI6 = TI2 - TI6
      T6 = TR3 + TR7
      T7 = TI3 + TI7
      TR7 = TR3 - TR7
      TI7 = TI3 - TI7
C
      TR0 = T0 + T4
      TI0 = T1 + T5
      TR2 = T0 - T4
      TI2 = T1 - T5
      TR1 = T2 + T6
      TI1 = T3 + T7
      TR3 = T2 - T6
      TI3 = T3 - T7
      T0 = TR4 - TI6
      T1 = TI4 + TR6
      T4 = TR4 + TI6
      T5 = TI4 - TR6
      T2 = TR5 - TI7
      T3 = TI5 + TR7
      T6 = TR5 + TI7
      T7 = TI5 - TR7
      BR0(J) = TR0 + TR1
      BI7(K) = TI0 + TI1
      BI6(K) = TR0 - TR1
      BR1(J) = TI1 - TI0
      BR2(J) = TR2 - TI3
      BI5(K) = TI2 + TR3
      BI4(K) = TR2 + TI3
      BR3(J) = TR3 - TI2
      PR = P7*(T2-T3)
      PI = P7*(T2+T3)
      BR4(J) = T0 + PR
      BI3(K) = T1 + PI
      BI2(K) = T0 - PR
      BR5(J) = PI - T1
      PR = -P7*(T6+T7)
      PI = P7*(T6-T7)
      BR6(J) = T4 + PR
      BI1(K) = T5 + PI
      BI0(K) = T4 - PR
      BR7(J) = PI - T5
```

```
100    CONTINUE
       JR = JR + 2
       JI = JI - 2
       IF (JI-JL) 110, 110, 120
110    JI = 2*JR - 1
       JL = JR
120   CONTINUE
      RETURN
      END
C
C----------------------------------------------------------------------
----
C SUBROUTINE:  R4SYN
C RADIX 4 SYNTHESIS
C----------------------------------------------------------------------
----
C
      SUBROUTINE R4SYN(INT, B0, B1, B2, B3)
      DIMENSION B0(2), B1(2), B2(2), B3(2)
      DO 10 K=1,INT
        T0 = B0(K) + B1(K)
        T1 = B0(K) - B1(K)
        T2 = B2(K) + B2(K)
        T3 = B3(K) + B3(K)
        B0(K) = T0 + T2
        B2(K) = T0 - T2
        B1(K) = T1 + T3
        B3(K) = T1 - T3
  10  CONTINUE
      RETURN
      END
C
C----------------------------------------------------------------------
----
C SUBROUTINE:  R8SYN
C RADIX 8 SYNTHESIS SUBROUTINE
C----------------------------------------------------------------------
----
C
      SUBROUTINE R8SYN(INT, NN, BR0, BR1, BR2, BR3, BR4, BR5, BR6,
 BR7,
     *    BI0, BI1, BI2, BI3, BI4, BI5, BI6, BI7)
      DIMENSION L(15), BR0(2), BR1(2), BR2(2), BR3(2), BR4(2),
 BR5(2),
     *    BR6(2), BR7(2), BI0(2), BI1(2), BI2(2), BI3(2), BI4(2),
     *    BI5(2), BI6(2), BI7(2)
      COMMON /CON1/ PII, P7, P7TWO, C22, S22, PI2
      EQUIVALENCE (L15,L(1)), (L14,L(2)), (L13,L(3)), (L12,L(4)),
     *    (L11,L(5)), (L10,L(6)), (L9,L(7)), (L8,L(8)), (L7,L(9)),
     *    (L6,L(10)), (L5,L(11)), (L4,L(12)), (L3,L(13)),
 (L2,L(14)),
     *    (L1,L(15))
      L(1) = NN/8
      DO 40 K=2,15
```

```
          IF (L(K-1)-2) 10, 20, 30
   10     L(K-1) = 2
   20     L(K) = 2
          GO TO 40
   30     L(K) = L(K-1)/2
   40   CONTINUE
        PIOVN = PII/FLOAT(NN)
        JI = 3
        JL = 2
        JR = 2
C
        DO 120 J1=2,L1,2
        DO 120 J2=J1,L2,L1
        DO 120 J3=J2,L3,L2
        DO 120 J4=J3,L4,L3
        DO 120 J5=J4,L5,L4
        DO 120 J6=J5,L6,L5
        DO 120 J7=J6,L7,L6
        DO 120 J8=J7,L8,L7
        DO 120 J9=J8,L9,L8
        DO 120 J10=J9,L10,L9
        DO 120 J11=J10,L11,L10
        DO 120 J12=J11,L12,L11
        DO 120 J13=J12,L13,L12
        DO 120 J14=J13,L14,L13
        DO 120 JTHET=J14,L15,L14
          TH2 = JTHET - 2
          IF (TH2) 50, 50, 90
   50     DO 60 K=1,INT
            T0 = BR0(K) + BR1(K)
            T1 = BR0(K) - BR1(K)
            T2 = BR2(K) + BR2(K)
            T3 = BR3(K) + BR3(K)
            T4 = BR4(K) + BR6(K)
            T6 = BR7(K) - BR5(K)
            T5 = BR4(K) - BR6(K)
            T7 = BR7(K) + BR5(K)
            PR = P7*(T7+T5)
            PI = P7*(T7-T5)
            TT0 = T0 + T2
            TT1 = T1 + T3
            T2 = T0 - T2
            T3 = T1 - T3
            T4 = T4 + T4
            T5 = PR + PR
            T6 = T6 + T6
            T7 = PI + PI
            BR0(K) = TT0 + T4
            BR1(K) = TT1 + T5
            BR2(K) = T2 + T6
            BR3(K) = T3 + T7
            BR4(K) = TT0 - T4
            BR5(K) = TT1 - T5
```

```
            BR6(K) = T2 - T6
            BR7(K) = T3 - T7
   6Ø     CONTINUE
          IF (NN-8) 12Ø, 12Ø, 7Ø
   7Ø     KØ = INT*8 + 1
          KL = KØ + INT - 1
          DO 8Ø K=KØ,KL
            T1 = BIØ(K) + BI6(K)
            T2 = BI7(K) - BI1(K)
            T3 = BIØ(K) - BI6(K)
            T4 = BI7(K) + BI1(K)
            PR = T3*C22 + T4*S22
            PI = T4*C22 - T3*S22
            T5 = BI2(K) + BI4(K)
            T6 = BI5(K) - BI3(K)
            T7 = BI2(K) - BI4(K)
            T8 = BI5(K) + BI3(K)
            RR = T8*C22 - T7*S22
            RI = -T8*S22 - T7*C22
            BIØ(K) = (T1+T5) + (T1+T5)
            BI4(K) = (T2+T6) + (T2+T6)
            BI1(K) = (PR+RR) + (PR+RR)
            BI5(K) = (PI+RI) + (PI+RI)
            T5 = T1 - T5
            T6 = T2 - T6
            BI2(K) = P7TWO*(T6+T5)
            BI6(K) = P7TWO*(T6-T5)
            RR = PR - RR
            RI = PI - RI
            BI3(K) = P7TWO*(RI+RR)
            BI7(K) = P7TWO*(RI-RR)
   8Ø     CONTINUE
          GO TO 12Ø
   9Ø     ARG = TH2*PIOVN
            C1 = COS(ARG)
            S1 = -SIN(ARG)
            C2 = C1**2 - S1**2
            S2 = C1*S1 + C1*S1
            C3 = C1*C2 - S1*S2
            S3 = C2*S1 + S2*C1
            C4 = C2**2 - S2**2
            S4 = C2*S2 + C2*S2
            C5 = C2*C3 - S2*S3
            S5 = C3*S2 + S3*C2
            C6 = C3**2 - S3**2
            S6 = C3*S3 + C3*S3
            C7 = C3*C4 - S3*S4
            S7 = C4*S3 + S4*C3
            INT8 = INT*8
            JØ = JR*INT8 + 1
            KØ = JI*INT8 + 1
            JLAST = JØ + INT - 1
            DO 1ØØ J=JØ,JLAST
```

```
K = KØ + J - JØ
TRØ = BRØ(J) + BI6(K)
TIØ = BI7(K) - BR1(J)
TR1 = BRØ(J) - BI6(K)
TI1 = BI7(K) + BR1(J)
TR2 = BR2(J) + BI4(K)
TI2 = BI5(K) - BR3(J)
TR3 = BI5(K) + BR3(J)
TI3 = BI4(K) - BR2(J)
TR4 = BR4(J) + BI2(K)
TI4 = BI3(K) - BR5(J)
TØ = BR4(J) - BI2(K)
T1 = BI3(K) + BR5(J)
TR5 = P7*(TØ+T1)
TI5 = P7*(T1-TØ)
TR6 = BR6(J) + BIØ(K)
TI6 = BI1(K) - BR7(J)
TØ = BR6(J) - BIØ(K)
T1 = BI1(K) + BR7(J)
TR7 = -P7*(TØ-T1)
TI7 = -P7*(T1+TØ)
TØ = TRØ + TR2
T1 = TIØ + TI2
T2 = TR1 + TR3
T3 = TI1 + TI3
TR2 = TRØ - TR2
TI2 = TIØ - TI2
TR3 = TR1 - TR3
TI3 = TI1 - TI3
T4 = TR4 + TR6
T5 = TI4 + TI6
T6 = TR5 + TR7
T7 = TI5 + TI7
TTR6 = TI4 - TI6
TI6 = TR6 - TR4
TTR7 = TI5 - TI7
TI7 = TR7 - TR5
BRØ(J) = TØ + T4
BIØ(K) = T1 + T5
BR1(J) = C1*(T2+T6) - S1*(T3+T7)
BI1(K) = C1*(T3+T7) + S1*(T2+T6)
BR2(J) = C2*(TR2+TTR6) - S2*(TI2+TI6)
BI2(K) = C2*(TI2+TI6) + S2*(TR2+TTR6)
BR3(J) = C3*(TR3+TTR7) - S3*(TI3+TI7)
BI3(K) = C3*(TI3+TI7) + S3*(TR3+TTR7)
BR4(J) = C4*(TØ-T4) - S4*(T1-T5)
BI4(K) = C4*(T1-T5) + S4*(TØ-T4)
BR5(J) = C5*(T2-T6) - S5*(T3-T7)
BI5(K) = C5*(T3-T7) + S5*(T2-T6)
BR6(J) = C6*(TR2-TTR6) - S6*(TI2-TI6)
BI6(K) = C6*(TI2-TI6) + S6*(TR2-TTR6)
BR7(J) = C7*(TR3-TTR7) - S7*(TI3-TI7)
BI7(K) = C7*(TI3-TI7) + S7*(TR3-TTR7)
```

```
 100      CONTINUE
          JR = JR + 2
          JI = JI - 2
          IF (JI-JL) 110, 110, 120
 110      JI = 2*JR - 1
          JL = JR
 120   CONTINUE
       RETURN
       END
C
C--------------------------------------------------------------------
----
C SUBROUTINE:  ORD1
C IN-PLACE REORDERING SUBROUTINE
C--------------------------------------------------------------------
----
C
       SUBROUTINE ORD1(M, B)
       DIMENSION B(2)
C
       K = 4
       KL = 2
       N = 2**M
       DO 40 J=4,N,2
         IF (K-J) 20, 20, 10
  10     T = B(J)
         B(J) = B(K)
         B(K) = T
  20     K = K - 2
         IF (K-KL) 30, 30, 40
  30     K = 2*J
         KL = J
  40   CONTINUE
       RETURN
       END
C
C--------------------------------------------------------------------
----
C SUBROUTINE:  ORD2
C IN-PLACE REORDERING SUBROUTINE
C--------------------------------------------------------------------
----
C
       SUBROUTINE ORD2(M, B)
       DIMENSION L(15), B(2)
       EQUIVALENCE (L15,L(1)), (L14,L(2)), (L13,L(3)), (L12,L(4)),
      *    (L11,L(5)), (L10,L(6)), (L9,L(7)), (L8,L(8)), (L7,L(9)),
      *    (L6,L(10)), (L5,L(11)), (L4,L(12)), (L3,L(13)),
 (L2,L(14)),
      *    (L1,L(15))
       N = 2**M
       L(1) = N
       DO 10 K=2,M
         L(K) = L(K-1)/2
```

```
  10   CONTINUE
       DO 20 K=M,14
         L(K+1) = 2
  20   CONTINUE
       IJ = 2
       DO 40 J1=2,L1,2
       DO 40 J2=J1,L2,L1
       DO 40 J3=J2,L3,L2
       DO 40 J4=J3,L4,L3
       DO 40 J5=J4,L5,L4
       DO 40 J6=J5,L6,L5
       DO 40 J7=J6,L7,L6
       DO 40 J8=J7,L8,L7
       DO 40 J9=J8,L9,L8
       DO 40 J10=J9,L10,L9
       DO 40 J11=J10,L11,L10
       DO 40 J12=J11,L12,L11
       DO 40 J13=J12,L13,L12
       DO 40 J14=J13,L14,L13
       DO 40 JI=J14,L15,L14
         IF (IJ-JI) 30, 40, 40
  30     T = B(IJ-1)
         B(IJ-1) = B(JI-1)
         B(JI-1) = T
         T = B(IJ)
         B(IJ) = B(JI)
         B(JI) = T
  40     IJ = IJ + 2
       RETURN
       END
C
C-----------------------------------------------------------------
----
C SUBROUTINE:  FFT842
C FAST FOURIER TRANSFORM FOR N=2**M
C COMPLEX INPUT
C-----------------------------------------------------------------
----
C
      SUBROUTINE FFT842(IN, N, X, Y)
C
C THIS PROGRAM REPLACES THE VECTOR Z=X+IY BY ITS  FINITE
C DISCRETE, COMPLEX FOURIER TRANSFORM IF IN=0.  THE INVERSE
 TRANSFORM
C IS CALCULATED FOR IN=1.  IT PERFORMS AS MANY BASE
C 8 ITERATIONS AS POSSIBLE AND THEN FINISHES WITH A BASE 4
 ITERATION
C OR A BASE 2 ITERATION IF NEEDED.
C
C THE SUBROUTINE IS CALLED AS SUBROUTINE FFT842 (IN,N,X,Y).
C THE INTEGER N (A POWER OF 2), THE N REAL LOCATION ARRAY X, AND
C THE N REAL LOCATION ARRAY Y MUST BE SUPPLIED TO THE SUBROUTINE.
```

```
C
      DIMENSION X(2), Y(2), L(15)
      COMMON /CON2/ PI2, P7
      EQUIVALENCE (L15,L(1)), (L14,L(2)), (L13,L(3)), (L12,L(4)),
     *    (L11,L(5)), (L1Ø,L(6)), (L9,L(7)), (L8,L(8)), (L7,L(9)),
     *    (L6,L(1Ø)), (L5,L(11)), (L4,L(12)), (L3,L(13)),
 (L2,L(14)),
     *    (L1,L(15))
C
C
C IW IS A MACHINE DEPENDENT WRITE DEVICE NUMBER
C
C     IW = I1MACH(2)
      IW=6
C
      PI2 = 8.*ATAN(1.)
      P7 = 1./SQRT(2.)
      DO 1Ø I=1,15
        M = I
        NT = 2**I
        IF (N.EQ.NT) GO TO 2Ø
  1Ø  CONTINUE
      WRITE (IW,9999)
9999  FORMAT (35H N IS NOT A POWER OF TWO FOR FFT842)
      STOP
  2Ø  N2POW = M
      NTHPO = N
      FN = NTHPO
      IF (IN.EQ.1) GO TO 4Ø
      DO 3Ø I=1,NTHPO
        Y(I) = -Y(I)
  3Ø  CONTINUE
  4Ø  N8POW = N2POW/3
      IF (N8POW.EQ.Ø) GO TO 6Ø
C
C RADIX 8 PASSES,IF ANY.
C
      DO 5Ø IPASS=1,N8POW
        NXTLT = 2**(N2POW-3*IPASS)
        LENGT = 8*NXTLT
        CALL R8TX(NXTLT, NTHPO, LENGT, X(1), X(NXTLT+1),
 X(2*NXTLT+1),
     *      X(3*NXTLT+1), X(4*NXTLT+1), X(5*NXTLT+1), X(6*NXTLT+1),
     *      X(7*NXTLT+1), Y(1), Y(NXTLT+1), Y(2*NXTLT+1),
 Y(3*NXTLT+1),
     *      Y(4*NXTLT+1), Y(5*NXTLT+1), Y(6*NXTLT+1), Y(7*NXTLT+1))
  5Ø  CONTINUE
C
C IS THERE A FOUR FACTOR LEFT
C
  6Ø  IF (N2POW-3*N8POW-1) 9Ø, 7Ø, 8Ø
C
C GO THROUGH THE BASE 2 ITERATION
```

```
C
C
  70  CALL R2TX(NTHPO, X(1), X(2), Y(1), Y(2))
      GO TO 90
C
C GO THROUGH THE BASE 4 ITERATION
C
  80  CALL R4TX(NTHPO, X(1), X(2), X(3), X(4), Y(1), Y(2), Y(3),
 Y(4))
C
  90  DO 110 J=1,15
        L(J) = 1
        IF (J-N2POW) 100, 100, 110
 100    L(J) = 2**(N2POW+1-J)
 110  CONTINUE
      IJ = 1
      DO 130 J1=1,L1
      DO 130 J2=J1,L2,L1
      DO 130 J3=J2,L3,L2
      DO 130 J4=J3,L4,L3
      DO 130 J5=J4,L5,L4
      DO 130 J6=J5,L6,L5
      DO 130 J7=J6,L7,L6
      DO 130 J8=J7,L8,L7
      DO 130 J9=J8,L9,L8
      DO 130 J10=J9,L10,L9
      DO 130 J11=J10,L11,L10
      DO 130 J12=J11,L12,L11
      DO 130 J13=J12,L13,L12
      DO 130 J14=J13,L14,L13
      DO 130 JI=J14,L15,L14
        IF (IJ-JI) 120, 130, 130
 120    R = X(IJ)
        X(IJ) = X(JI)
        X(JI) = R
        FI = Y(IJ)
        Y(IJ) = Y(JI)
        Y(JI) = FI
 130    IJ = IJ + 1
      IF (IN.EQ.1) GO TO 150
      DO 140 I=1,NTHPO
        Y(I) = -Y(I)
 140  CONTINUE
      GO TO 170
 150  DO 160 I=1,NTHPO
        X(I) = X(I)/FN
        Y(I) = Y(I)/FN
 160  CONTINUE
 170  RETURN
      END
C
C------------------------------------------------------------------
----
```

```
C SUBROUTINE:  R2TX
C RADIX 2 ITERATION SUBROUTINE
C-----------------------------------------------------------------------
----
C
      SUBROUTINE R2TX(NTHPO, CRØ, CR1, CIØ, CI1)
      DIMENSION CRØ(2), CR1(2), CIØ(2), CI1(2)
      DO 1Ø K=1,NTHPO,2
        R1 = CRØ(K) + CR1(K)
        CR1(K) = CRØ(K) - CR1(K)
        CRØ(K) = R1
        FI1 = CIØ(K) + CI1(K)
        CI1(K) = CIØ(K) - CI1(K)
        CIØ(K) = FI1
  1Ø  CONTINUE
      RETURN
      END
C
C-----------------------------------------------------------------------
----
C SUBROUTINE:  R4TX
C RADIX 4 ITERATION SUBROUTINE
C-----------------------------------------------------------------------
----
C
      SUBROUTINE R4TX(NTHPO, CRØ, CR1, CR2, CR3, CIØ, CI1, CI2,
 CI3)
      DIMENSION CRØ(2), CR1(2), CR2(2), CR3(2), CIØ(2), CI1(2),
 CI2(2),
     *    CI3(2)
      DO 1Ø K=1,NTHPO,4
        R1 = CRØ(K) + CR2(K)
        R2 = CRØ(K) - CR2(K)
        R3 = CR1(K) + CR3(K)
        R4 = CR1(K) - CR3(K)
        FI1 = CIØ(K) + CI2(K)
        FI2 = CIØ(K) - CI2(K)
        FI3 = CI1(K) + CI3(K)
        FI4 = CI1(K) - CI3(K)
        CRØ(K) = R1 + R3
        CIØ(K) = FI1 + FI3
        CR1(K) = R1 - R3
        CI1(K) = FI1 - FI3
        CR2(K) = R2 - FI4
        CI2(K) = FI2 + R4
        CR3(K) = R2 + FI4
        CI3(K) = FI2 - R4
  1Ø  CONTINUE
      RETURN
      END
C
C-----------------------------------------------------------------------
----
```

```
C SUBROUTINE:  R8TX
C RADIX 8 ITERATION SUBROUTINE
C-----------------------------------------------------------------------
----
C
      SUBROUTINE R8TX(NXTLT, NTHPO, LENGT, CRØ, CR1, CR2, CR3, CR4,
     *    CR5, CR6, CR7, CIØ, CI1, CI2, CI3, CI4, CI5, CI6, CI7)
      DIMENSION CRØ(2), CR1(2), CR2(2), CR3(2), CR4(2), CR5(2),
 CR6(2),
     *    CR7(2), CI1(2), CI2(2), CI3(2), CI4(2), CI5(2), CI6(2),
     *    CI7(2), CIØ(2)
      COMMON /CON2/ PI2, P7
C
      SCALE = PI2/FLOAT(LENGT)
      DO 3Ø J=1,NXTLT
        ARG = FLOAT(J-1)*SCALE
        C1 = COS(ARG)
        S1 = SIN(ARG)
        C2 = C1**2 - S1**2
        S2 = C1*S1 + C1*S1
        C3 = C1*C2 - S1*S2
        S3 = C2*S1 + S2*C1
        C4 = C2**2 - S2**2
        S4 = C2*S2 + C2*S2
        C5 = C2*C3 - S2*S3
        S5 = C3*S2 + S3*C2
        C6 = C3**2 - S3**2
        S6 = C3*S3 + C3*S3
        C7 = C3*C4 - S3*S4
        S7 = C4*S3 + S4*C3
        DO 2Ø K=J,NTHPO,LENGT
          ARØ = CRØ(K) + CR4(K)
          AR1 = CR1(K) + CR5(K)
          AR2 = CR2(K) + CR6(K)
          AR3 = CR3(K) + CR7(K)
          AR4 = CRØ(K) - CR4(K)
          AR5 = CR1(K) - CR5(K)
          AR6 = CR2(K) - CR6(K)
          AR7 = CR3(K) - CR7(K)
          AIØ = CIØ(K) + CI4(K)
          AI1 = CI1(K) + CI5(K)
          AI2 = CI2(K) + CI6(K)
          AI3 = CI3(K) + CI7(K)
          AI4 = CIØ(K) - CI4(K)
          AI5 = CI1(K) - CI5(K)
          AI6 = CI2(K) - CI6(K)
          AI7 = CI3(K) - CI7(K)
          BRØ = ARØ + AR2
          BR1 = AR1 + AR3
          BR2 = ARØ - AR2
          BR3 = AR1 - AR3
          BR4 = AR4 - AI6
          BR5 = AR5 - AI7
          BR6 = AR4 + AI6
```

```
          BR7 = AR5 + AI7
          BIØ = AIØ + AI2
          BI1 = AI1 + AI3
          BI2 = AIØ - AI2
          BI3 = AI1 - AI3
          BI4 = AI4 + AR6
          BI5 = AI5 + AR7
          BI6 = AI4 - AR6
          BI7 = AI5 - AR7
          CRØ(K) = BRØ + BR1
          CIØ(K) = BIØ + BI1
          IF (J.LE.1) GO TO 1Ø
          CR1(K) = C4*(BRØ-BR1) - S4*(BIØ-BI1)
          CI1(K) = C4*(BIØ-BI1) + S4*(BRØ-BR1)
          CR2(K) = C2*(BR2-BI3) - S2*(BI2+BR3)
          CI2(K) = C2*(BI2+BR3) + S2*(BR2-BI3)
          CR3(K) = C6*(BR2+BI3) - S6*(BI2-BR3)
          CI3(K) = C6*(BI2-BR3) + S6*(BR2+BI3)
          TR = P7*(BR5-BI5)
          TI = P7*(BR5+BI5)
          CR4(K) = C1*(BR4+TR) - S1*(BI4+TI)
          CI4(K) = C1*(BI4+TI) + S1*(BR4+TR)
          CR5(K) = C5*(BR4-TR) - S5*(BI4-TI)
          CI5(K) = C5*(BI4-TI) + S5*(BR4-TR)
          TR = -P7*(BR7+BI7)
          TI = P7*(BR7-BI7)
          CR6(K) = C3*(BR6+TR) - S3*(BI6+TI)
          CI6(K) = C3*(BI6+TI) + S3*(BR6+TR)
          CR7(K) = C7*(BR6-TR) - S7*(BI6-TI)
          CI7(K) = C7*(BI6-TI) + S7*(BR6-TR)
          GO TO 2Ø
1Ø        CR1(K) = BRØ - BR1
          CI1(K) = BIØ - BI1
          CR2(K) = BR2 - BI3
          CI2(K) = BI2 + BR3
          CR3(K) = BR2 + BI3
          CI3(K) = BI2 - BR3
          TR = P7*(BR5-BI5)
          TI = P7*(BR5+BI5)
          CR4(K) = BR4 + TR
          CI4(K) = BI4 + TI
          CR5(K) = BR4 - TR
          CI5(K) = BI4 - TI
          TR = -P7*(BR7+BI7)
          TI = P7*(BR7-BI7)
          CR6(K) = BR6 + TR
          CI6(K) = BI6 + TI
          CR7(K) = BR6 - TR
          CI7(K) = BI6 - TI
2Ø      CONTINUE
3Ø    CONTINUE
      RETURN
      END
```

```
*****************************************************************
                          CCXTRA.FOR
*****************************************************************

      SUBROUTINE RP(X,Y,R,TH,N,ID)
C   REC - POLAR CONVERSION
C     ID NOT =1 ==> R->P
C     ID =1     ==> P->R
      REAL*8 X(16384),Y(16384),R(16384),TH(16384)
      REAL*8 DSQRT,DATAN2,DCOS,DSIN
      IF(ID.EQ.1)GO TO 10
      DO 5 I=1,N
      R(I)=DSQRT(X(I)*X(I)+Y(I)*Y(I))
      TH(I)=0.D0
      IF(Y(I).EQ.0.D0.AND.X(I).EQ.0.D0)GO TO 5
      TH(I)=DATAN2(Y(I),X(I))
    5 CONTINUE
      RETURN
   10 CONTINUE
      DO 15 I=1,N
      X(I)=R(I)*DCOS(TH(I))
   15 Y(I)=R(I)*DSIN(TH(I))
      RETURN
      END
C
C
C
      SUBROUTINE MR1DF(LOG2N,X,Y,SIGN)
C     FORTRAN VERSION
C     MIXED RADIX FOURIER TRANSFORM
      INTEGER LOG2N
      REAL*8 X(16384),Y(16384)
      DIMENSION S(13),U(13)
C
      INTEGER J1,J2,J3,J4,JT,N,M4
      REAL*8 ARG,C1,C2,C3,S1,S2,S3,R1,R2,R3,R4,R5,R6,R7,R8,T
      REAL*8 DCOS,DSIN
C
C
C
      INTEGER A,B,C,D,E,F,G,H,I,J,K,L,M,
     1     BS,CS,DS,ES,FS,GS,HS,IS,JS,KS,LS,MS,
     2     AL,BL,CL,DL,EL,FL,GL,HL,IL,JL,KL,ML,S,U
      EQUIVALENCE
 (BS,S(2)),(CS,S(3)),(DS,S(4)),(ES,S(5)),(FS,S(6)),
     1
 (GS,S(7)),(HS,S(8)),(IS,S(9)),(JS,S(10)),(KS,S(11)),(LS,S(12)),
     2
 (MS,S(13)),(AL,U(1)),(BL,U(2)),(CL,U(3)),(DL,U(4)),(EL,U(5)),
     3
 (FL,U(6)),(GL,U(7)),(HL,U(8)),(IL,U(9)),(JL,U(10)),(KL,U(11)),
     4 (LL,U(12)),(ML,U(13))
```

```
C
      N=2**LOG2N
      IF (SIGN) 8000,8000,8002
8000  DO 8001 I=1,N
8001  Y(I)=-Y(I)
8002  CONTINUE
        IF (LOG2N-1) 500,500,501
 501   CONTINUE
      DO 400 K=2,LOG2N,2
      M=2**(LOG2N-K)
      M4=4*M
      DO 300 J=1,M
        ARG=6.28315D0*DBLE(FLOAT(J-1)/FLOAT(M4))
      C1=DCOS(ARG)
      S1=DSIN(ARG)
      C2=C1*C1-S1*S1
      S2=C1*S1+C1*S1
      C3=C2*C1-S2*S1
      S3=C2*S1+S2*C1
      DO 200 I=M4,N,M4
      J1=I+J-M4
      J2=J1+M
      J3=J2+M
      J4=J3+M
      R1=X(J1)+X(J3)
      R2=X(J1)-X(J3)
      R3=Y(J1)+Y(J3)
      R4=Y(J1)-Y(J3)
      R5=X(J2)+X(J4)
      R6=X(J2)-X(J4)
      R7=Y(J2)+Y(J4)
      R8=Y(J2)-Y(J4)
      X(J1)=R1+R5
      Y(J1)=R3+R7
        IF(ARG) 101,100,101
 101    CONTINUE
      X(J3)=(R2+R8)*C1+(R4-R6)*S1
      Y(J3)=(R4-R6)*C1-(R2+R8)*S1
      X(J2)=(R1-R5)*C2+(R3-R7)*S2
      Y(J2)=(R3-R7)*C2-(R1-R5)*S2
      X(J4)=(R2-R8)*C3+(R4+R6)*S3
      Y(J4)=(R4+R6)*C3-(R2-R8)*S3
       GO TO 200
 100   CONTINUE
       X(J3)=R2+R8
       Y(J3)=R4-R6
       X(J2)=R1-R5
       Y(J2)=R3-R7
       X(J4)=R2-R8
       Y(J4)=R4+R6
 200   CONTINUE
 300   CONTINUE
 400   CONTINUE
```

```
500    CONTINUE
          ITEST=LOG2N-(LOG2N/2*2)
         IF(ITEST) 701,700,701
 701    CONTINUE
       DO 600 I=1,N,2
       R1=X(I)+X(I+1)
       R2=X(I)-X(I+1)
       R3=Y(I)+Y(I+1)
       R4=Y(I)-Y(I+1)
       X(I)=R1
       Y(I)=R3
       X(I+1)=R2
       Y(I+1)=R4
600    CONTINUE
700    CONTINUE
       MS=N/2
       ML=N
       DO 800 K=2,12
       J=14-K
       S(J)=1
       U(J)=S(J+1)
          IF(S(J+1)-1)    7701,7701,7700
7700      S(J)=S(J+1)/2
7701    CONTINUE
800    CONTINUE
       AL=BS
       JT=0
       DO 900 A=1,AL
       DO 900 B=A,BL,BS
       DO 900 C=B,CL,CS
       DO 900 D=C,DL,DS
       DO 900 E=D,EL,ES
      DO 900 F=E,FL,FS
       DO 900 G=F,GL,GS
       DO 900 H=G,HL,HS
       DO 900 I=H,IL,IS
       DO 900 J=I,JL,JS
       DO 900 K=J,KL,KS
       DO 900 L=K,LL,LS
       DO 900 M=L,ML,MS
       JT=JT+1
            IF(JT-M) 900,900,901
 901        CONTINUE
       T=X(JT)
       X(JT)=X(M)
       X(M)=T
       T=Y(JT)
       Y(JT)=Y(M)
       Y(M)=T
 900   CONTINUE
       RETURN
       END
```

```
****************************************************************
                        SHIFTF.FOR
****************************************************************


      SUBROUTINE SHIFTF(F,N)
C...
C...  SHIFTS PHASE OF ARRAY SO THAT ANGLE OF FIRST ELEMENT IS
 ZERO
C...
C...  GIVEN THE FIRST ELEMENT a+ib AND ANY ELEMENT X+iY
C...
C...                                   *
C...                                *  *
 *
C...     sqrt(a**2+b**2) *             *                        *
 *
C...                  *                * b   and            *
 * Y
C...               *                   *                 *
 *
C...            * angle                *             * ANGLE
 *
C...          ********************           ********************
*
C...                  a                                 X
C...
C...  SHIFT THE PHASE BY angle
C...                                         *
C...                                         *
C...                                *        *
C...                        *                * Y1
C...             ANGLE-angle                 *
C...          **************************
C...                      X1
C...
C...          SIN(angle)=b/sqrt(a**2+b**2)
C...          COS(angle)=a/sqrt(a**2+b**2)
C...
C...          | X1 |   | COS(angle)    SIN(sngle)|  | X |
C...          |    | = |                         |  |   |
C...          | Y1 |   |-SIN(angle)    COS(angle)|  | Y |
C...  SUCH THAT:
C...          X1= [a/sqrt(a**2+b**2)]*X + [b/sqrt(a**2+b**2)]*Y
C...          Y1=-[b/sqrt(a**2+b**2)]*X + [a/sqrt(a**2+b**2)]*Y
C...

C*****************************************************************
*********
      DIMENSION F(1)
C...
      A=F(1)
      B=F(2)
```

```
      AMULT=A/SQRT(A**2+B**2)
      BMULT=B/SQRT(A**2+B**2)
C...
      DO 100 I=1,N+2,2
      X=F(I)
      Y=F(I+1)
      F(I)=AMULT*X+BMULT*Y
      F(I+1)=-BMULT*X+AMULT*Y
100   CONTINUE
      RETURN
      END
```

Appendix C

PATTERN RECOGNITION
SCATTER PLOT PRODUCTION

Appendix C

PATTERN RECOGNITION SCATTER PLOT PRODUCTION

Start with several sampled data files that represent the data from which pattern similarities are to be extracted. Each segment of data to be analyzed is labeled. Label files must be created with the ILS command $ LBF 0, which is similar to the manner in which sampled data files are created with the $ FIL 0 command as in this example. Use of a command file is suggested when many label files are to be created.

To label the segments, use ILS command $ LBA. Data segments must have been previously selected by use of one of the ILS commands $ DSP or $ CUR so that this information is in the user's common file. A label file contains a series of labels, each one of which is a two-line record of ASCII characters that describes a segment of interest in a sampled data file. Next, use ILS command $ QUR to extract signal features. Because ILS command $ QUR restricts the user to 32 frequency spectra, a local version was used that selects 32 evenly spaced spectra from the output of a 2048-point FFT. To use QUR (or XUR), first designate the label file that contains pointers to the area of interest in the signal. $ QUR output will be in the form of feature record files.

Using ILS command $ SME, create output feature record files which contain mean vector and eigen records. Since $ SME is now restricted to 20 elements in analysis, first extract the 20 most significant elements in the output records using ILS command $ MRE. Use the same elements for all records analyzed. If you have used QUR on single data samples, collect the samples in two or more record files using $ TRE for analysis with $ SME. In this study, ILS command $ SME 6 was used. One output record file is produced by $ SME containing the mean and eigen vector records. Next, using the files input to $ SME and the output file of $ SME, perform a principal component analysis of those input files. $ PCO outputs a record file for each input file. Plot the results with $ PLR.

Examples are given on the following pages.

Example:
Ten sampled data files exist which are numbered 1405 through 1495 in steps of 10.
Create the label file(s).

```
$ LBF O <cr>
 OPTIONS TO MANIPULATE LABEL FILES:
 ENTER FILENAME [,DK] TO SELECT A FILE
 ENTER C <RETURN>     TO CREATE A FILE
 ENTER I <RETURN>     TO SET THE INITIALS
 ENTER   <RETURN>     TO EXIT
->C
 ENTER FILENAME [,DK]
->TAM1
 DRB0:[ILSMGR.DGN]TAM1.LAB      LABEL DATA
     0 LABELS,  INITIALS:
 PRIMARY LABEL FILE

 OPTIONS TO MANIPULATE LABEL FILES:
 ENTER FILENAME [,DK] TO SELECT A FILE
 ENTER C <RETURN>     TO CREATE A FILE
 ENTER I <RETURN>     TO SET THE INITIALS
 ENTER   <RETURN>     TO EXIT
->I
 ENTER THE INITIALS
->TAM
 DRB0:[ILSMGR.DGN]TEST2.LAB      LABEL DATA
     0 LABELS,  INITIALS:  TAM
 PRIMARY LABEL FILE

 OPTIONS TO MANIPULATE LABEL FILES:
 ENTER FILENAME [,DK] TO SELECT A FILE
 ENTER C <RETURN>     TO CREATE A FILE
 ENTER I <RETURN>     TO SET THE INITIALS
 ENTER   <RETURN>     TO EXIT
-><cr>
```

In like manner create the other nine label files.

Use of a command file is suggested when many label files are to be created.

Label the segments, having previously indicated the range by the cursor command or by the $ DSP range.
Point to the label file by issuing the command $ LBF TAM1.

```
 DRB0:[ILSMGR.ILS]TAM1.LAB      LABEL DATA
     1 LABELS,  INITIALS:
 PRIMARY LABEL FILE
$ LBA

 FIELD-1 IS OBTAINED FROM THE FILE HEADER, @=ABORT
 ENTER FIELD-2;FIELD-3;FIELD-4;FIELD-5;FIELD-6
->TAMTEST
```

List the label file

```
$ LLA
 DRB0:[ILSMGR.ILS]TAM1.LAB              22-MAY-84          PAGE    1
```

```
 *****  LABEL       1  ***************************************************
 TAMTEST                   ;  ;        ;     ;                    ;
  1;   1;    1;2Ø48; 2ØØØ;DRAØ:[ILSMGR.ILS]WD14Ø5.        ;27-JAN-84;TAM
Label the rest of the files.
Extract the features
$ XUR
 ENTER MODE, EXTRACTION CODE AND NUMBER OF FEATURES
 MODE:     M = MEAN FRAMES
           C = CONSECUTIVE FRAMES
 FEATURE: 1 = AUTOREGRESSIVE COEFFICIENTS
           2 = REFLECTION COEFFICIENTS
           3 = AUTOCORRELATION COEFFICIENTS
           4 = FREQUENCY SPECTRA
           5 = LP  CEPSTRAL COEFFICIENTS
           6 = MEL CEPSTRAL COEFFICIENTS
           7 = RESONANCE FREQ AND BAND (PEAK PICK)
           8 = RESONANCE FREQ AND BAND (ROOT SOLVE)
           9 = NORMALIZED RESONANCE FREQ AND BAND (ROOT SOLVE)
->4
           EXTRACTION CODE =     4
 USING LABEL  FILE....DRBØ:[ILSMGR.ILS]TXM1.LAB
 USING RECORD FILE....DRBØ:[ILSMGR.ILS]WD1ØØØ.
 LEVEL  1
 FIELD-1 IS OBTAINED FROM THE FILE HEADER, @=ABORT
 ENTER FIELD-2;FIELD-3;FIELD-4;FIELD-5;FIELD-6 FOR SEARCH
 ENTER <RETURN> TO START SEARCH
->
 DRBØ:[ILSMGR.ILS]WD1ØØØ.         RECORD     1 STORED
And so on with the rest of the files.
Using $ MRE move the first twenty elements in each file
to another file
$ MRE
 MOVE FEATURE RECORDS FROM A FILE AND

 EXTRACT SELECTED ITEMS AND ELEMENTS FROM RECORDS
 ===========================

 ENTER...ELEMENT NUMBER(S) TO EXTRACT FROM EACH ITEM
         USE <ALL> <RETURN> FOR ALL ELEMENTS
         USE <F>   <RETURN> TO FINISH
         USE <A>   <RETURN> TO ABORT .
         MAXIMUM OF 12 ENTRIES/LINE, NEGATIVE IMPLIES INCLUSIVE
->1,-2Ø
->F
 ENTER...ITEM NUMBER(S) TO EXTRACT FROM EACH RECORD
         USE <ALL> <RETURN> FOR ALL ITEMS
         USE <F>   <RETURN> TO FINISH
         USE <A>   <RETURN> TO ABORT
         MAXIMUM OF 12 ENTRIES/LINE, NEGATIVE IMPLIES INCLUSIVE
->ALL
 ENTER...RECORD NUMBER(S) OF EACH RECORD TO TRANSFER
         USE <ALL> <RETURN> FOR ALL RECORDS
         USE <F>   <RETURN> TO FINISH
         USE <T>   <RETURN> TO TEST ON ELEMENTS (MAX=1Ø)
         USE <SK>  NO. REC  <RETURN> TO SKIP RECORDS
```

```
          USE <AV>  NO. REC  <RETURN> TO AVERAGE RECORDS
          USE <AVI> NO. REC  <RETURN> TO AVERAGE ITEMS IN RECORDS
          USE <A>   <RETURN> TO ABORT
          MAXIMUM OF 12 ENTRIES/LINE, NEGATIVE IMPLIES INCLUSIVE
->ALL
 ENTER...FILE NO., DISK NO. OF INPUT FILE
->1Ø1Ø
 ENTER...FILE NO., DISK NO. OF OUTPUT FILE
->1Ø21
 DRBØ:[ILSMGR.ILS]WD1Ø21.         RECORD     1 STORED
And so on with the rest of the files.
Collect the records in two files for analysis.
$ FIL 1Ø21
 DRBØ:[ILSMGR.ILS]WD1Ø21.         RECORD DATA
    12 DK BLKS,    1 RECORDS
 PRIMARY FILE
$ FIL S1Ø31
 DRBØ:[ILSMGR.ILS]WD1Ø31.         DOES NOT EXIST
 SECONDARY FILE
$ OPN S2
$ TRE 1,1,5,1,2,5
 USING DRBØ:[ILSMGR.ILS]WD1Ø21.
 DRBØ:[ILSMGR.ILS]WD1Ø31.         RECORD     1 STORED
 USING DRBØ:[ILSMGR.ILS]WD1Ø22.
 DRBØ:[ILSMGR.ILS]WD1Ø31.         RECORD     2 STORED
 USING DRBØ:[ILSMGR.ILS]WD1Ø23.
 DRBØ:[ILSMGR.ILS]WD1Ø31.         RECORD     3 STORED
 USING DRBØ:[ILSMGR.ILS]WD1Ø24.
 DRBØ:[ILSMGR.ILS]WD1Ø31.         RECORD     4 STORED
 USING DRBØ:[ILSMGR.ILS]WD1Ø25.
 DRBØ:[ILSMGR.ILS]WD1Ø31.         RECORD     5 STORED
 USING DRBØ:[ILSMGR.ILS]WD1Ø26.
 DRBØ:[ILSMGR.ILS]WD1Ø32.         RECORD     1 STORED
 USING DRBØ:[ILSMGR.ILS]WD1Ø27.
 DRBØ:[ILSMGR.ILS]WD1Ø32.         RECORD     2 STORED
 USING DRBØ:[ILSMGR.ILS]WD1Ø28.
 DRBØ:[ILSMGR.ILS]WD1Ø32.         RECORD     3 STORED
 USING DRBØ:[ILSMGR.ILS]WD1Ø29.
 DRBØ:[ILSMGR.ILS]WD1Ø32.         RECORD     4 STORED
 USING DRBØ:[ILSMGR.ILS]WD1Ø3Ø.
 DRBØ:[ILSMGR.ILS]WD1Ø32.         RECORD     5 STORED
$ FIL 1Ø31
 DRBØ:[ILSMGR.ILS]WD1Ø31.         RECORD DATA
    12 DK BLKS,    5 RECORDS
 PRIMARY FILE
$ FIL S1Ø33
 DRBØ:[ILSMGR.ILS]WD1Ø33.         DOES NOT EXIST
 SECONDARY FILE
$ OPN S3
$ SME 6,2
 DO YOU WANT VARIANCE RATIOS PRINTED? (Y OR N)
->N
 DRBØ:[ILSMGR.ILS]WD1Ø33.         RECORD     4 STORED
$ FIL S1Ø34
```

```
 DRBØ:[ILSMGR.ILS]WD1Ø34.         RECORD DATA
    12 DK BLKS,     Ø RECORDS
 SECONDARY FILE
$ PCO 2,1Ø33
 USER$$DISK:[ILSMGR.ILS]WD1Ø34.   RECORD      5 STORED

 USER$$DISK:[ILSMGR.ILS]WD1Ø35.   RECORD      5 STORED
Plot the resaults using $ PLR
$ PLR
 ILS PLOTTING ROUTINE
     UP TO 2Ø INPUT FILES
     UP TO 1Ø24 POINTS PLOTTED
 ENTER X-COMPONENT
->1
 ENTER Y-COMPONENT
->2
 ENTER FILE NO., DISK NO.
 AND NO. CONSECUTIVE FILES
  DEFAULTS TO DRBØ:[ILSMGR.ILS]WD1Ø31.
->1Ø34,,2
 SYMBOLS USED ARE A-B
 READ IN ANOTHER FILE? (Y,N)
->N
 FILES BEING READ
 ENTER SYMBOL TO PLOT, OR ENTER <ALL>
->ALL
 CC = CARTESIAN
 SL = SEMI-LOG
 LL = LOG-LOG
->CC
 AUTOMATIC SCALING? (Y,N)
->Y
 GRID? (Y,N)
->N
 ENTER OPTION
 E=EXIT
 N[S,A]=NEW FILES
        [START,APPEND]
 C=COMPONENTS
 M[I]=MARK [IDENTIFY]
 R=REASSIGN
 G=GRID
 S=SCALING
 T=TYPE OF PLOT
 D=DATA TO PLOT
 P[N]=PLOT [NO ERASE]
 F[A,E,L]N1,N2=
    FACTOR ANAL
 H=HARD COPY
->E
```

END

FILMED

3-85

DTIC